A Survey of Group Key Distribution Schemes With Self-Healing Property

Tomasz Rams and Piotr Pacyna, *Member, IEEE*

***Abstract*—Secure key distribution schemes for group communications allow to establish a secure multicast communication between a group manager and group members through an unreliable broadcast channel. The article classifies, analyzes and compares the most significant key distribution schemes, by looking at the selective key distribution algorithms, at the predistributed secret data management, and at the self-healing mechanisms. It reviews polynomial-based algorithms, exponential arithmetic based algorithms, hash-based techniques, and others. Attention is paid to the self-healing property, which permits group members to recover missing session keys from the recent key distribution broadcast message, without any additional interaction with the group manager.**

***Index Terms*—Security, cryptographic protocols, multicast communication, key distribution, self-healing.**

I. Introduction

SELF-healing group key distribution schemes has recently received a lot of attention from the researchers, as a method enabling large and dynamic groups of users to establish group keys over unreliable network for secure multicast communication. In such schemes, time is divided into epochs called sessions. At the beginning of each session, a Group Manager transmits some broadcast message, in order to provide a common key to each member of the group. Every user, belonging to the group, computes the group key using the message and some private information. The main property of the scheme is that, if some broadcast message gets lost, then users are still capable of recovering the group key for that session by using the message they received at the beginning of a previous session and the message they will receive at the beginning of a subsequent one, without requesting additional transmission from the Group Manager. This approach decreases the workload on the Group Manager and reduces network traffic as well as the risk of user exposure through traffic analysis.

A. Our contribution

~~The objective of this paper is to provide the Reader with a comprehensive review of the development in the area of self-healing key distribution schemes. It should be especially useful as a reference for new researchers, as it identifies basic building blocks of the scheme and describes in details all~~ major types of existing solutions. ~~It also contains a thorough security and efficiency analysis of each solution, and points out issues not identified by the Authors in the original papers.~~ We decided to include in this paper a description of a few broken schemes, to explain in detail insecure mechanisms proposed in the literature, which were reused in the later solutions, based on the incorrect security analysis provided in the original papers. We believe, that this can help researchers avoid repeating mistakes in their future schemes.

The article reviews the most significant self-healing group key distribution schemes, and it gives insight into open research problems in this area. Functionality of the scheme is decomposed into three separate aspects, namely: *selective key distribution mechanism*, *predistributed secret data management* and *self-healing mechanism*, which are used to classify and compare schemes. Each aspect of the scheme is discussed separately, based on the most significant solutions proposed in the literature. Finally, comprehensive comparison of several representative schemes, based on the set of universal, quantitative metrics, is presented.

Two articles, [1] and [2], reviewing self-healing key distribution schemes have recently appeared in the literature. Tian et al. in [1] provides a survey of available solutions, that is focused on the possible scheme extensions, such as *sponsorization* or *mutual-healing*. The Authors propose classification of schemes based on the applied cryptographic primitives. However, many schemes use several primitives, thus making such classification difficult. In [2] the author analyzes practicality of self-healing key distribution schemes in resource-constrained Wireless Sensor Networks (WSN). The conclusion is that almost none of the schemes is suitable for large scale WSN in real-world applications. This review is focused on the scheme performance in terms of the communication and storage overhead. It does not describe or analyze details of the applied algorithms.

In our paper, we introduce a three-dimensional classification, based on each aspect of the scheme separately, which allows for more flexibility. The classification facilitates the comparison of schemes, because it is easier to identify similarities in mechanisms reused in several solutions. Furthermore, we provide a description of some important techniques, such as exponential arithmetic algorithms or different types of self-healing mechanisms, which have been omitted in the other articles. We also introduce some novel observations and technical details, such as possible attacks on the access polynomial approach. The security and efficiency analysis shows that a number of claims about security of the schemes referred to in the surveys require clarification.

Manuscript received 26 September 2011; revised 16 June 2012. This work has been funded by the Polish Ministry of Science and Higher Education under grant No. N N517 228135 and R&D project No. OR00011912.

T. Rams and P. Pacyna are with AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Krakow, Poland (e-mail: {trams,pacyna}@kt.agh.edu.pl).

Digital Object Identifier 10.1109/SURV.2012.081712.00144

1553-877X/13/$31.00 © 2013 IEEE

B. Applications

Self-healing group key distribution schemes can be used in multicast networks with centralized management, which are established over an unreliable broadcast channel, such as machine-to-machine systems, embedded and sensor networks, cellular networks and wireless networks. They can be also applied in the broadcast transmission systems, including cable and satellite television, pay-per-view TV and information services. Packet losses are expected in these applications, so they can significantly benefit from the self-healing mechanism.

The schemes appear to be useful in settings, in which session keys need to be used for a short time-period, in order to reduce the amount of ciphertext available to an adversary, or need to be updated due to frequent changes in the group structure. Public safety and military applications can benefit from reliability and strong security of the self-healing group key distribution schemes.

C. Organization of the paper

The paper is organized as follows. Section II describes basic idea and the most important characteristics of the self-healing approach. Section III identifies essential elements of the scheme. In Sections IV through VI analyses of each element are presented, based on the most significant solutions available in the literature. Comprehensive comparison of several representative schemes is given in Section VII. The article is concluded in Section VIII.

II. Self-healing approach

Self-healing schemes for group key distribution have been an active research area in recent years. In this section we introduce the basic idea and the most important characteristics of the self healing approach to the group key distribution.

A. Overview of the network model

Self-healing group key distribution schemes can be used in various network scenarios, thus, to make their analysis and comparison easier, we introduce an abstract model of the network in which these schemes are applicable.

The network consists of a single Group Manager (GM) and a finite universe of User Nodes (U). Group Manager is a resource rich node with high computational power, large memory space, and unlimited energy resources. User nodes, on the other hand, have limited computational power, limited memory, and limited energy resources. GM communicates with nodes in U through an unreliable broadcast channel. She transmits broadcast messages which are received by all users. Because of nodes mobility and channel communications errors, some messages can be lost. Message retransmission should be avoided, if possible, since it is costly and requires feedback connection from receiver nodes to GM, which may not always be available.

The main goal is to establish secure multicast communication between GM and members of a group of nodes $G \subseteq U$, which is a subset of U. Group G is dynamic, user nodes can join and leave. Communications security is achieved by message encryption and authentication using shared symmetric secret group key K. A shared key is convenient, but it can be disclosed by nodes leaving the group, or by group members intercepted by an adversary. To achieve high security level the key shall be changed frequently throughout the group lifetime. To do so, secure group key distribution mechanism, with GM acting as a Trust Anchor, is needed for key replacement.

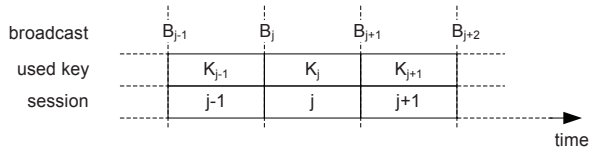

Fig. 1. Group lifetime divided into sessions

A prospective group key distribution scheme should satisfy the following requirements:

- *Authorization*. The scheme should prevent adversaries or unauthorized user nodes, which are not in G, from learning the group key.
- *Key freshness*. Key distribution scheme has to provide fresh keys.
- *Efficiency*. Group key distribution scheme shall be efficient with respect to communication, computational, memory, and energy cost. It shall take into account user nodes limitations.
- *Scalability*. Network size usually ranges from dozens to hundreds of thousands nodes, so the scheme has to be scalable to be practically applied.
- *Communications model*. The scheme should be applicable to the network model described in this section.

B. Group lifetime divided into sessions

Common group key is frequently updated to ensure secure multicast communication. Group lifetime is dived into epochs called *sessions*, single key instance is valid only throughout one session (Fig. 1). Group membership can change between consecutive sessions (denoted by G_j in session j). At the beginning of session j, GM distributes a new session key K_j to nodes in G_j. Session duration is determined by the GM. It can vary over time, depending on security policy, group membership changes and nodes' behavior. Session key changes have to be performed, with some predefined minimum frequency to protect the system from cryptanalysis attacks. Moreover, to effectively remove a node U_i from multicast group G_j, who is willing to leave, or is forced to leave, a new session must begin and nodes from G_{j+1} shall start protecting group communication using a new K_{j+1}, which is not accessible to U_i. Thus, the choice of session length is a tradeoff between key distribution cost in terms of communication and computational overhead, and the required security level.

C. Broadcast distribution of session keys

At the beginning of each session j, GM transmits a broadcast message B_j to distribute session key K_j to all nodes in G_j. Since B_j is broadcast, it can be received by any node in U, including the ones which are not authorized to obtain K_j. Thus, to prevent from unauthorized access, K_j has to

be hidden in B_j using some additional masking data. This is usually achieved in three phases:

1) *Set Up:* GM generates personal key S_i for each node belonging to a multicast group, and distributes it to U_i over secure channel. Personal key S_i is secret information used only by U_i to recover any future session keys from broadcast messages. Nodes can be equipped with personal keys before network deployment, if set of potential group members can be predetermined, or they can obtain their personal keys from GM when they join multicast group.
2) *Broadcast:* GM creates message B_j from K_j in a way fulfilling the following requirements:
   a) There is an efficient algorithm η, which for all $i : U_i \in G_j$, can be used to recover K_j knowing S_i, that is: $K_j = \eta(B_j, S_i)$.
   b) There is no computationally viable algorithm ς, which for any set of nodes $R \subset U \setminus G_j$, can be used to recover K_j knowing personal keys of all nodes in R, that is: $K_j = \varsigma(B_j, \{S_l\}_{l:U_l \in R})$ is infeasible.
3) *Session key recovery:* Every member $U_j \in G_j$ recovers key K_j from received message B_j using her personal key S_i, by calculating $K_j = \eta(B_j, S_i)$.

D. User join and revocation

Group G is dynamic, which means user nodes can spontaneously join and leave it. A new node U_i joining the group receives from GM her personal key S_i valid in consecutive sessions $(r, \ldots, s)$, where r is a session in which the user joins the group, and $s+1$ is a session in which she is supposed to leave the group. U_i must not be able to obtain session keys, which are used in sessions before r and after s. U_i can also be forcibly removed from multicast group before S_i expires. To revoke node U_i in session $l : r < l \leq s$, GM has to construct all messages $(B_l, \ldots, B_s)$ in a way rendering S_i useless.

E. Stateless key distribution

Self-healing key distribution schemes are stateless, that is they always permit group members $U_i \in G_j$ to obtain valid session key K_j from the last broadcast message B_j, also when they miss some previous key distribution messages. This property is usually achieved by storing all the necessary and up-to-date group state information at GM and transmitting it in every broadcast B_j. User nodes do not update their personal keys upon receiving broadcast messages, so the keys stay valid even if they lack some messages. Nodes, which recover from temporary communication loss, can obtain current session key, immediately after receiving the next broadcast message, and become fully functional group members.

F. Key recovery through self-healing

The purpose of self-healing is to add some additional information to message B, which would allow user nodes to recover keys from previous sessions lost due to communication errors. User nodes shall be able to recover lost session keys on their own, without any additional interaction with GM. This is achieved by combining information from any message B_l preceding the lost packet B_j with information from any message B_r following it. Formally, given any l, r, such that $l < r$, there is an efficient algorithm ζ which for all $j : l < j < r$ can be used by node $U_i \in G_l \cap G_j \cap G_r$ to recover K_j from known B_l and B_r, that is $K_j = \zeta(B_l, B_r, S_i)$[1]. It should be noted that self-healing mechanism allows to recover only past session keys. Users, who have lost the most recent message B_j, are not able to obtain current session key K_j until the next session $j+1$, in which they correctly receive message B_{j+1}. Thus, self-healing mechanism does not guarantee availability of the most recent keys.

It is worth recalling here, that user nodes can obtain the most recent session key K_j despite previous communication problems, upon receiving correct B_j, even without self-healing mechanism, because this is guaranteed by the stateless property of key distribution scheme. In systems, where only the most recent session key is used to protect multicast group communication, self-healing property does not yield significant profits. It can be only used by user nodes to validate and decrypt messages collected in previous sessions, but not processed yet due to lack of appropriate session keys. However, with slight modification of key distribution scheme described so far, self-healing property can become really valuable.

As it was suggested in [4], introduction of a delay of d sessions between distribution of a session key, and usage of this key for the protection of multicast group communication, allows to improve availability of the currently used keys at the cost of key freshness. At the beginning of session j every node $U_i \in G_j$ recovers K_j from the received B_j and stores it in a key vector $(K_{j-d+1}, \ldots, K_j)$. During session j, key K_{j-d} is used to protect group communication (Fig. 2). Due to the buffering of recent d keys, user nodes can seamlessly handle loss of up to $(d-1)$ consecutive broadcast messages, since knowledge of messages B_{j-d} and B_j allows to recover all session keys $(K_{j-d+1}, \ldots, K_{j-1})$ using self-healing property. Unfortunately, delay between session key distribution and session key usage has an impact on node revocation. Node revocation performed by GM is effective only after d sessions, i.e. when group communication is protected by the session key, which is no longer accessible for the revoked node.

G. Formal definition of self-healing group key distribution

To clarify scheme analysis, we introduce a general model of a self-healing group key distribution scheme. It is a generalized version of the model proposed by Blundo et al. in [5], so that it can cover both unconditionally secure and computationally secure schemes.

Let us assume that m is the number of sessions supported by the scheme, and t is the maximum number of users, that can be revoked during the scheme lifetime[2]. Self-healing group

[1]This is the original definition of the self-healing property, introduced by Staddon et al. in [3], however in some techniques presented in Section VI recovery of lost key K_j can be performed based on the single message B_r and also requirement that $U_i \in G_l \cap G_j \cap G_r$ may be significantly relaxed, so that U_i must only be a member of the group G_j.

[2]Note, that there exist solutions with unlimited number of sessions, or unlimited number of users that can be revoked, but they should be considered as a limit case, with $m = \infty$, or $t = |U|$ respectively.

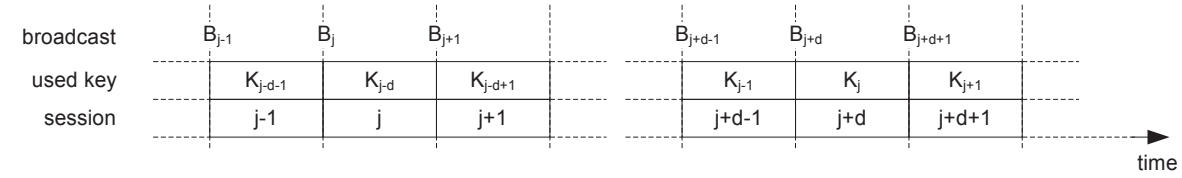

Fig. 2. Usage of distributed session key delayed by d sessions

key distribution scheme is formally defined in Definition 1. Desired security properties of the scheme are described in Definition 2.

Definition 1 (Self-healing group key distribution scheme): Let U be the universe of users of a network, let m be the maximum number of sessions, and let t be the maximum number of users that can be revoked by GM. The scheme is a *self-healing group key distribution scheme* if the following conditions are true:

1) *It is a session key distribution scheme*:
   a) For each member $U_i \in G_j$, the key K_j is determined by B_j and S_i. In other words, there is an efficient algorithm η, which for all $i : U_i \in G_j$, can be used to compute $K_j = \eta(B_j, S_i)$.
   b) What users learn from the broadcast B_j and their own personal key cannot be determined from the broadcasts or personal keys alone. In other words, there is no computationally viable algorithm θ, which, for any $j = 1, \ldots, m$ can be used to compute $K_j = \theta(B_1, \ldots, B_m)$ or $K_j = \theta(S_{G_{<1,m>}})$, where $S_{G_{<1,m>}}$ denotes the set of personal keys of all users in $G_1 \cup \ldots \cup G_m$.
2) *It has t-revocation capability* — for each session j, let $R_{<1,j>} = R_j \cup \ldots \cup R_1$, such that $|R_{<1,j>}| \leq t$, the group manager GM can generate a broadcast message B_j such that all revoked users in $R_{<1,j>}$ cannot recover K_j. In other words, there is no computationally viable algorithm ς, which can be used to compute $K_j = \varsigma(B_1, \ldots, B_j, S_{R_{<1,j>}})$, where $S_{R_{<1,j>}}$ denotes personal keys of all users in $R_{<1,j>}$.
3) *It has self-healing capability* — every $U_i \in G_l$, who has not been revoked after session l and before session r, from broadcast messages B_l and B_r, where $1 \leq l < r \leq m$, can recover all keys K_j, for $j = l, \ldots, r$. In other words, there is an efficient algorithm ζ which for all $j = l, \ldots, r$ can be used to calculate $K_j = \zeta(B_l, B_r, S_i)$.

Definition 2 (Scheme security properties): Let U be the universe of users of a network, let m be the maximum number of sessions, and let t be the security parameter. Self healing group key distribution scheme is considered secure, if the following properties are satisfied:

1) *t-forward secrecy* — given any set $R_{<1,j>} \subseteq U$ of users revoked up to and including session j, such that $|R_{<1,j>}| \leq t$, it is computationally infeasible for the users $U_i \in R_{<1,j>}$ colluding together to recover any of subsequent session keys $K_j, \ldots, K_m$.
2) *t-backward secrecy* — given any set $J_{<j+1,m>} \subseteq U$ of users who joined the group after session j, such that $|J_{<j+1,m>}| \leq t$, it is computationally infeasible for the users $U_i \in J_{<j+1,m>}$ colluding together to recover any of past session keys $K_1, \ldots, K_j$.
3) *t-collusion resistance* — given any set $R_{<1,l>} \subseteq U$ of users revoked from the group up to and including session l, and any set $J_{<r+1,m>} \subseteq U$ of users who join the group after session r, such that $|R_{<1,l>} \cup J_{<r+1,m>}| \leq t$, it is computationally infeasible for a colluding coalition $R_{<1,l>} \cup J_{<r+1,m>}$, to recover any of session keys $K_l, \ldots, K_r$.

III. Essential elements of self-healing key distribution schemes

We identified three elements, or building blocks, of self-healing key distribution scheme which can be used to classify and compare the existing solutions:

1) selective key distribution mechanism,
2) predistributed secret data management,
3) self-healing mechanism.

For simplicity, in selective key distribution mechanism we consider a single session case, and then, in predistributed secret data management, we extend our considerations to the multiple sessions. Finally, in self-healing mechanism we describe how to add self-healing property to the scheme. Thus, to maintain comprehensibility, our notation slightly differs during the description of each element of the self-healing key distribution scheme.

Selective key distribution mechanism is an algorithm used to distribute single session key K to all authorized users $U_i \in G$, using broadcast media. It specifies how to calculate broadcast data chunk b, in such a way that all users $U_i \in G$ will be able to obtain session key K from b using their predistributed secret bit strings s_i, while the revoked users $U_r \in R$ will not be able to recover any information about K. Selective access to the broadcast session key is based on the fact that every user $U_i \in U$ is equipped with different secret predistributed data, which is used during calculation of K. Since in the selective key distribution mechanism only a single session is considered and all variables are associated with the same session, we use the following notation:

- K is a session key,
- b is a broadcast data chunk,
- s_i is an instance of secret predistributed data assigned to user U_i,
- G is a set of authorized group members,
- R is a set of revoked users.

Life-cycle of the group key distribution schemes usually consists of a large number of sessions, and selective key distribution has to be repeated in each session. Therefore, every user has to store her own secret data, needed for all expected

sessions, which is delivered using a secure communication channel. This is usually done offline, before deployment of the node, or when a new node joins the group. Predistributed secret data management specifies how secret data stored by users is organized, and how it is used for the selective key distribution. Structure of the predistributed data determines user storage requirements, and hence limitations of the scheme lifetime.

The basic idea of self-healing technique is to add some additional information to the broadcast message, which would allow user nodes to recover previous session keys, lost due to communication errors. User nodes shall be able to recover lost session keys on their own, without any additional interaction with GM. Self-healing mechanism specifies how broadcast messages are constructed and how users can use the additional data from them, to recover lost session keys. In the description of the predistributed secret data management and self-healing mechanism we use the same notation, that is:

- K_j is a session key distributed in session j,
- b_j is a data chunk broadcast to selectively distribute K_j,
- B_j is a message broadcast in session j, which may include multiple data chunks b_i and some additional data needed for self-healing mechanism, for example $B_j = [b_1, \ldots, b_j]$,
- $s_{i,j}$ is an instance of secret predistributed data used by user U_i to recover K_j from b_j,
- S_i is an entire set of secret predistributed data delivered to U_i, which usually consists of multiple instances $s_{i,j}$, for example $S_i = [s_{i,1}, s_{i,2}, \ldots, s_{i,m}]$
- G_j is a set of authorized group members in session j,
- R_j is a set of users revoked in session j.

In this paragraph we describe the most significant approaches proposed in literature for each element of the self-healing scheme.

IV. Selective key distribution mechanism

Selective key distribution mechanism is an algorithm used by GM to deliver a single session key K to a group G using broadcast media. It specifies how to calculate a publicly known bit string b, in such a way that all users $U_i \in G$ will be able to obtain session key K from b using their own predistributed secret bit strings s_i, while the revoked users $U_r \in R$ will not be able to recover any information about the key K. There exist four main classes of the selective key distribution algorithms (*SKD*) applied in self-healing schemes:

1) polynomial based algorithms,
2) exponential arithmetic based algorithms,
3) vector space secret sharing based algorithms,
4) bilinear pairings based algorithms.

We present each approach, along with the most significant algorithms available in the literature, and then we point out its main characteristics. The definition of the selective key distribution algorithm consists of three elements:

- predistributed users data,
- broadcast public data,
- session key calculation procedure.

We use these elements in a repetitive fashion so as to allow for comparison of various algorithms.

A. Polynomial based algorithms

Polynomial based algorithms are a group of SKD algorithms utilizing arithmetic on polynomials defined over a finite field F_q. Polynomials are used to decrease the amount of data which has to be transmitted in order to securely deliver the session key to authorized users only. Predistributed user secret bit strings are not chosen randomly, but instead, they are related to each other, which allows for a trade-off between security of such algorithm and communication overhead. All computations take place in F_q, where q is a large prime. Let:

- $I_U = \{x_i \in F_q\}_{U_i \in U}$ be the set of all indices assigned to the universe of users, where x_i is assigned to user U_i,
- $I_R = \{x_i \in F_q\}_{U_i \in R}$ be the set of indices assigned to users which are revoked,
- $I_G = \{x_i \in F_q\}_{U_i \in G}$ be the set of indices assigned to authorized members of the group G,
- $H(\cdot)$ be the entropy function.

1) Bivariate polynomial secret sharing SKD: Bivariate polynomial secret sharing SKD algorithm was proposed by Staddon et al. in the paper introducing the idea of self-healing schemes [3]. It is an unconditionally secure selective key distribution algorithm allowing to revoke up to t users. The algorithm is an extension of Naor-Pinkas algorithm [6].

- *User U_i predistributed data:* $s_i = [N, x_i, s(x_i, x_i)]$, where $N \in F_q$ is a randomly chosen variable, $x_i \in I_U$ is a random index assigned to user U_i, and $s(x, y) \in F_q[x, y]$ is a random bivariate polynomial of degree t in each variable.
- *Broadcast public data:* $b = [s(N, x) + K, \{(w_l, s(w_l, x))\}_{w_l \in W}]$, where $W = \{w_1, \ldots, w_t\} \subseteq F_q$ is a set of random distinct indices such that $|W| = t$, $I_R \subseteq W$, $I_G \bigcap W = \emptyset$ and $N \notin W$.
- *Session key calculation procedure for user U_i:*
  1) Recover polynomial $s(x, x_i)$ by Lagrange interpolation based on t points $\{(w_l, s(w_l, x_i))\}_{w_l \in W}$ obtained by evaluation of polynomials, received in b, at $x = x_i$, and one extra point $(x_i, s(x_i, x_i))$ known from predistributed data.
  2) Calculate session key K by evaluating $s(N, x) + K$ at $x = x_i$ and by subtracting $s(x, x_i)|_{x=N}$, that is: $K = (s(N, x) + K)|_{x=x_i} - s(x, x_i)|_{x=N}$.

In order to be able to obtain from broadcast data chunk b any information about session key K, user U_i has to know at least one point of polynomial $s(N, x)$. The user recovers polynomial $s(x, x_i)$ by Lagrange interpolation, and evaluates it at $x = N$ to get point $(x_i, s(N, x_i))$. To correctly interpolate polynomial of degree t, U_i has to know at least $t + 1$ distinct points of it. Since $I_R \subseteq W$, all users belonging to R have only access to t distinct points $\{(w_l, s(w_l, x_i))\}_{w_l \in W}$ and none of them is able to recover any information about session key from b. A collaborating group of such users is not able to recover any additional information about K.

It should be noted that during the distribution of key K, bivariate polynomial $s(x, y)$ is recovered by all users $U_i \in G$, so it cannot be securely reused in any later session. After obtaining K, user U_i is able to calculate polynomial $s(N, x)$ and then interpolate entire polynomial $s(x, y)$ using $s(N, x)$, and t polynomials $\{s(w_l, x)\}_{w_l \in W}$.

The algorithm allows to revoke up to t users, which is determined by the degree of polynomial $s(x, y)$. Predistributed data size is $3 \log q$ and broadcast data size is $(t^2+3t+1) \log q$. Bivariate polynomial secret sharing SKD algorithm is rather complicated and relatively expensive in terms of broadcast data size and computational cost at each U_i. It has been replaced by a simpler and more efficient univariate polynomial secret sharing algorithm.

2) Univariate polynomial secret sharing SKD: Blundo et al. in [5] first applied univariate polynomial secret sharing SKD algorithm to provide a selective session key distribution in self-healing scheme. The algorithm was based on techniques developed in [6], [7]. It uses univariate t-degree polynomial to provide user revocation.

- *User U_i predistributed data:* $s_i = [x_i, s(x_i)]$, where $x_i \in I_U$ is a random index assigned to user U_i, and $s(x) \in F_q[x]$ is a random polynomial of degree t.
- *Broadcast public data:* $b = [s(0) + K, \{(w_l, s(w_l))\}_{w_l \in W}]$, where $W = \{w_1, \ldots, w_t\} \subseteq F_q$ is a set of random distinct indices such that $|W| = t$, $I_R \subseteq W$, $I_G \bigcap W = \emptyset$ and $0 \notin W$.
- *Session key calculation procedure for user U_i:*
  1) Recover polynomial $s(x)$ by Lagrange interpolation based on t points $\{(w_l, s(w_l))\}_{w_l \in W}$ received in b, and one point $(x_i, s(x_i))$ from predistributed data.
  2) Calculate session key $K = (s(0) + K) - s(x)|_{x=0}$.

In this algorithm, session key K is hidden by a secret value $s(0)$. User U_i has to interpolate polynomial $s(x)$ to calculate $s(0)$ and obtain K. To correctly interpolate polynomial of degree t, user U_i has to know at least $t + 1$ distinct points on it. All users together belonging to R have only access to t distinct points $\{(w_l, s(w_l))\}_{w_l \in W}$, because $I_R \subseteq W$ and points from their predistributed data are already included in b. Thus, none of them is able to recover any information about the session key from b. Also a collaborating group of such users is not able to recover any additional information about K. More precisely, for any set R, such that $|R| \leq t$, equation $H(K|b, \{s_i\}_{U_i \in R}) = H(K)$ holds, which means that algorithm is unconditionally secure and t-conspiracy resistant.

Note that during the distribution of a single key K, polynomial $s(x)$ is recovered by all users $U_i \in G$. Thus, in this algorithm, like in bivariate polynomial secret sharing algorithm, a given polynomial $s(x)$ can be securely used only in one session.

Univariate polynomial secret sharing SKD algorithm allows to revoke up to t users, a value of which is determined by the degree of polynomial $s(x)$. Predistributed data size is $2 \log q$ and broadcast data size is $(2t + 1) \log q$. The algorithm provides the same functionality and security level as the bivariate polynomial secret sharing SKD, but it is much simpler and more efficient in terms of broadcast data size and computational cost. It is applied in later self-healing schemes proposed by Dutta et al.—in construction 2 of [8], construction 1 of [9] and construction 1 of [10].

3) Revocation polynomial SKD: Revocation polynomial SKD algorithm was first proposed by Liu et al. in [11]. Later, Hong et al. [12] proposed a simplified version of the algorithm, which is more efficient in terms of broadcast data size. This version is one of the most popular selective key distribution algorithms used in self-healing schemes. We present both versions of the algorithm, to fully explain characteristics of the revocation polynomial approach.

Algorithm proposed in construction 3 of [11] is as follows:

- *User U_i predistributed data:* $s_i = [x_i, h(x_i), f(x_i)]$, where $x_i \in I_U$ is a random index assigned to user U_i, $h(x) \in F_q[x]$ is a random polynomial of degree $2t$, and $f(x) \in F_q[x]$ is a random polynomial of degree t.
- *Broadcast public data:* $b = [I_R, P(x) = r(x)p(x)+h(x), Q(x) = q(x) + f(x)]$, where $r(x)$ is a revocation polynomial $r(x) = \prod_{x_i \in I_R}(x - x_i)$, $|I_R| \leq t$, $p(x) \in F_q[x]$ is a random t-degree polynomial, and $q(x) = K - p(x)$.
- *Session key calculation procedure for user U_i:*
  1) Recover $p(x_i)$ by evaluating polynomial $P(x)$ at $x = x_i$, and subtracting $h(x_i)$, and by dividing the result by $r(x_i)$, that is $p(x_i) = \frac{P(x)|_{x=x_i} - h(x_i)}{r(x_i)}$
  2) Recover $q(x_i) = Q(x)|_{x=x_i} - f(x_i)$
  3) Calculate session key $K = p(x_i) + q(x_i)$

The SKD algorithm of Liu is tightly coupled with the self-healing mechanism applied in the scheme. Session key K is represented by two random polynomials $q(x)$ and $p(x)$, such that $q(x) = K - p(x)$. To obtain the K, user U_i has to know the value of each polynomial for at least one argument $x = x_i$. Selective access to the key share $p(x)$ is achieved by combination of revocation polynomial $r(x)$ and masking polynomial $h(x)$. The masking polynomial ensures that users can only evaluate product of polynomials $r(x) \cdot p(x)$ at the x for which they know value of $h(x)$, that is at $x = x_i$, using $h(x_i)$ known from their predistributed data. For all revoked users $U_i \in R$, revocation polynomial $r(x)$ evaluates to 0 at $x = x_i$, so they are not able to recover any information about $p(x)$, because $P(x)|_{x=x_i} = h(x_i)$. Since the knowledge of $q(x_i)$ without knowing $p(x_i)$ does not give any information about K, session key distribution is unconditionally secure. Also a collaborating group of revoked users is not able to recover any additional information about session key. More precisely, for any set R, such that $|R| \leq t$, equation $H(K|b, \{s_i\}_{U_i \in R}) = H(K)$ holds, which means that algorithm is unconditionally secure and t-conspiracy resistant.

The algorithm of Liu allows to revoke up to t users, which is determined by the degree of masking polynomial $h(x)$. Degree of $r(x)$ depends on the number of the revoked users, and in the worst case scenario it is equal to t, so the degree of the polynomial calculated as a product of $p(x)$ and $r(x)$ is up to $2t$. Thus, the masking polynomial has to be of degree $2t$, to completely hide polynomial $r(x) \cdot p(x)$.

Predistributed data size in this algorithm is $3 \log q$ and broadcast data size is up to $(4t + 2) \log q$, where q is an order of the field F_q over which polynomials are defined. The algorithm is significantly less efficient than univariate polynomial secret sharing algorithm, but it should be noted that it actually performs selective distribution of different points on t-degree polynomial $p(x)$ instead of the distribution of single session key K. This is a result of incorrect requirement introduced in the first articles [3], [11] about self-healing key distribution schemes, that predistributed users data must not be disclosed to other users during session key distribution process. The

requirement was rejected by Blundo et al. in [5], and it was shown in [13] by using information theoretic arguments, that it can not be satisfied.

Hong et al. [12] proposed a simplified version of the revocation polynomial SDK algorithm. It is similar to the algorithm of Liu, but instead of the selective distribution of t-degree polynomial $p(x)$, it performs distribution of the session key K. In the rest of the paper this version of the algorithm will just be referred to as the revocation polynomial SDK algorithm.

- *User U_i predistributed data:* $s_i = [x_i, h(x_i)]$, where $x_i \in I_U$ is a random index assigned to user U_i, and $h(x) \in F_q[x]$ is a random polynomial of degree t.
- *Broadcast public data:* $b = [I_R, P(x) = r(x) \cdot K + h(x)]$, where $r(x)$ is called revocation polynomial, and is defined as $r(x) = \prod_{x_i \in I_R}(x - x_i)$.
- *Session key calculation procedure for user U_i:*
  1) Calculate K by evaluating polynomial $P(x)$ at $x = x_i$, subtracting $h(x_i)$, and by dividing the result by $r(x_i)$, that is $K = \frac{P(x)|_{x=x_i} - h(x_i)}{r(x_i)}$.

In this algorithm, a selective access to the session key K is achieved by the combination of revocation polynomial $r(x)$ and masking polynomial $h(x)$. Masking polynomial ensures that users can only evaluate polynomial $r(x) \cdot K$ at x, for which they know value of $h(x)$. For all revoked users $U_i \in R$, revocation polynomial $r(x)$ evaluates to 0 at $x = x_i$, so they are not able to recover any information about K, because $P(x)|_{x=x_i} = h(x_i)$. Also a collaborating group of revoked users is not able to recover any additional information about K. More precisely, for any set R, such that $|R| \leq t$, equation $H(K|b, \{s_i\}_{U_i \in R}) = H(K)$ holds, which means that algorithm is unconditionally secure and t-conspiracy resistant.

Note that during the distribution of a single key K, masking polynomial $h(x)$ is disclosed to all users $U_i \in G$, as opposed to algorithm of Liu, where only partial information about $h(x)$ was revealed. Thus, in this algorithm, a single polynomial $h(x)$ can be securely used only in one session.

The algorithm allows to revoke up to t users, which is determined by the degree of masking polynomial $h(x)$. Degree of polynomial $r(x) \cdot K$ depends on the number of revoked users, and is up to t, so masking polynomial $h(x)$ of the degree t is sufficient to hide it completely. In this algorithm the size of the predistributed data is $2 \log q$ and the size of the broadcast data is $(|R|+t+1) \log q$. For the worst case scenario, where $|R| = t$, the cost of the algorithm in terms of broadcast data size is the same as the cost of the univariate polynomial secret sharing SKD. Otherwise, it is less expensive. Moreover, it is also possible to compress I_R across the sessions, as it was described in [12], because users revoked in one session usually remain revoked in later sessions.

Revocation polynomial SKD algorithm has become one of the most popular selective key distribution algorithm applied in self-healing schemes, because of its simplicity and efficiency. It was employed by Dutta et al. in [14], construction 1 of [8], [15], constructions 2 and 3 of [9], and constructions 2 and 3 of [10]. It was also adopted by other researchers—Yuan et al. [16], Du et al. [17] and Han et al. [18].

4) Access polynomial SKD: Access polynomial SKD approach has recently received attention as it was expected to overcome shortcomings of previous SKD algorithms. Unfortunately, it turned out, that most of the algorithms based on this approach are insecure or inefficient. We briefly introduce the most significant algorithms and point out their weaknesses. We use common notation, which slightly differs from notations used in original papers.

The concept of access polynomial was first introduced by Zou et al. in [19]. It is based on polynomial $a(x)$ which is constructed in such a way, that for all authorized users it evaluates to 1, but for all other users it evaluates to some random value. This can be seen as the concept opposite to the revocation polynomial approach. The details of the algorithm are as follows.

- *User U_i predistributed data:* $s_i = [x_i, h(x_i)]$, where $x_i \in I_U$ is a random, *secret* index assigned to user U_i, and $h(x) \in F_q[x]$ is a random polynomial of degree t.
- *Broadcast public data:* $b = [P(x) = a(x)p(x) + h(x), Q(x) = q(x) + h(x)]$, where $a(x)$ is called access polynomial, and is defined as $a(x) = (x-r)\prod_{x_i \in I_G}(x - x_i) + 1$, $r \in F_q \setminus I_U$ is a randomly selected value, $p(x) \in F_q[x]$ is a random t-degree polynomial, and $q(x) = K - p(x)$.
- *Session key calculation procedure for user U_i:*
  1) Recover $p(x_i)$ by evaluating polynomial $P(x)$ at $x = x_i$, and by subtracting $h(x_i)$, that is $p(x_i) = P(x)|_{x=x_i} - h(x_i)$. The calculation gives correct result, because for each $x_i \in I_G$, $a(x_i) = 1$, so $P(x_i) = p(x_i) + h(x_i)$.
  2) Recover $q(x_i) = Q(x)|_{x=x_i} - h(x_i)$.
  3) Calculate session key $K = p(x_i) + q(x_i)$.

The authors suggest that the algorithm does not disclose any information about $h(x)$ or about the set of secret authorized indices I_G, during the distribution of session key, and as such $h(x)$ and I_G can be securely reused in later sessions. We prove that this is not true, and that the algorithm is insecure.

Note that the degree of polynomial $a(x)$ depends on the number of authorized users, and is equal to $|G| + 1$, so the degree of product of polynomials $a(x)$ and $p(x)$ is $|G|+t+1$. As can be seen, polynomial $h(x)$ of degree t cannot hide it completely. Let's calculate how many unknown variables are included in b, and how many independent linear equations can be constructed based on the content of b, to assess the leak of secret information during a single key distribution. The total number of variables in b is $|G|+2t+4$, and the total number of independent linear equations is $|G| + 2t + 3$. It means that any user or group of users, which is able to construct at least one additional independent linear equation, will be able to recover all information about K, $h(x)$ and I_G. Even predistributed user data can be used as additional information, to break this algorithm. Hence, it cannot be securely used in self-healing schemes.

Access polynomial approach was adopted by Tian et al. [20], who created a simpler and more efficient algorithm.

- *User U_i predistributed data:* $s_i = [x_i, h(x_i)]$, where $x_i \in I_U$ is a random, *secret* index assigned to user U_i, and $h(x) \in F_q[x]$ is a random polynomial of degree t.

- *Broadcast public data:* $b = [P(x) = a(x) \cdot K + h(x)]$, where $a(x)$ is called access polynomial, and is defined as $a(x) = (x - r) \prod_{x_i \in I_G} (x - x_i) + 1$, and $r \in F_q \setminus I_U$ is a randomly selected value.
- *Session key calculation procedure for user U_i :*
  1) Calculate session key $K = P(x)|_{x=x_i} - h(x_i)$. The calculation gives correct result, because for each $x_i \in I_G$, $a(x_i) = 1$, and as a result $P(x_i) = K + h(x_i)$.

In [20] the authors also claim, like in the algorithm by [19], that the algorithm does not disclose any information about $h(x)$ or about the set of secret authorized indices I_G, during the distribution of session key, and as such $h(x)$ and I_G can be securely reused in later sessions. This is not true, and can be proved in the analogous way as it was done for the previous scheme. We focus on the other issue of this algorithm, which may be even more painful in specific settings.

The degree of polynomial $a(x)$ is $|G|+1$, so for a group size greater or equal to t, masking polynomial $h(x)$ does not hide it completely. Let $(\alpha_{|G|+1}, \ldots, \alpha_0)$ be a set of coefficients of polynomial $a(x) = \alpha_{|G|+1} x^{|G|+1} + \alpha_{|G|} x^{|G|} + \ldots + \alpha_1 x + \alpha_0$, and let $(h_t, \ldots, h_0)$ be a set of coefficients of polynomial $h(x) = h_t x^t + h_{t-1} x^{t-1} + \ldots + h_1 x + h_0$. Coefficients of polynomial $P(x)$, which are broadcast in b are equal to $(p_{|G|+1} = \alpha_{|G|+1} K, \ldots, p_{t+1} = \alpha_{t+1} K, p_t = \alpha_t K + h_t, \ldots, p_0 = \alpha_0 K + h_0)$. Since, by definition, $\alpha_{|G|+1}$ is equal to 1, anyone who receives b is able to recover session key $K = p_{|G|+1}$.

A similar algorithm with slight modification was recently proposed by Dutta et al. in [21]. In this algorithm no masking polynomial was used, which may be seen as a special case of Tian's algorithm, where $h(x) = 0$. This modification makes the algorithm vulnerable to the above issue, for any size of the authorized group G. Anybody is able to recover session key K from the data included in b. Thus, the algorithm actually does not perform selective key distribution, but it gives the same effect as a simple broadcast of the session key.

Yuan et al. in [22], [23] proposed an access polynomial SKD algorithm which is unconditionally secure, but can be applied only for specific group settings, where the number of revoked users is much larger than the number of authorized users. This characteristic of the group was called *limited group membership property*. Security of the algorithm is achieved by limiting the maximum size of G to the degree of masking polynomial, and by a slight change in definition and usage of access polynomial.

- *User U_i predistributed data:* $s_i = [x_i, h(x_i)]$, where $x_i \in I_U$ is a random, *secret* index assigned to user U_i, and $h(x) \in F_q[x]$ is a random polynomial of degree t.
- *Broadcast public data:* $b = [P(x) = a(x) + K + h(x)]$, where $a(x)$ is called access polynomial, and is defined as $a(x) = r \prod_{x_i \in W} (x - x_i)$, $r \in F_q \setminus I_U$ is a randomly selected value, W is a set of random distinct indices such that $W \subset F_q$, $|W| = t$, $I_G \subseteq W$ and $(I_U \setminus I_G) \bigcap W = \emptyset$.
- *Session key calculation procedure for user U_i:*
  1) Calculate session key K by evaluating polynomial $P(x)$ at $x = x_i$, and by subtracting $h(x_i)$, that is $K = P(x)|_{x=x_i} - h(x_i)$. The calculation gives correct result, because for each $x_i \in I_G$, $a(x_i) = 0$, and as a result $P(x_i) = K + h(x_i)$.

Selective access to K is achieved by the combination of access polynomial $a(x)$ and masking polynomial $h(x)$. Masking polynomial ensures that users can only evaluate polynomial $P(x)$ at the x for which they know value of $h(x)$. For all unauthorized users $U_i \in U \setminus G$, access polynomial $a(x)$ evaluates to some unknown, random value, so they are not able to recover any information about K, because all values of K are equally probable. Also a collaborating group of the revoked users, of size up to t is not able to recover any additional information about K. More precisely, for any set of the revoked users R, such that $|R| \leq t$, equation $H(K|b, \{s_i\}_{U_i \in R}) = H(K)$ holds, which means that the algorithm is unconditionally secure and t-conspiracy resistant.

In [22] it was suggested that $h(x)$ is not disclosed during key distribution, and that it can be securely reused in several sessions. Actually, it is not true, because $h(x)$ has to be changed in each session, as it is done in [23], to protect secrecy of I_G. If the same polynomial $h(x)$ is used multiple times, partial information about I_G is disclosed, which may be exploited to obtain an unauthorized access to K.

The algorithm allows to revoke any number of users. Unlike revocation polynomial SDK, here, the degree of masking polynomial $h(x)$ is not a limitation for maximum number of the revoked users. However, a collaborating group of more than t revoked users is still able to interpolate polynomial $h(x)$, and then recover K and secret set I_G. Thus, access polynomial SKD algorithm does not limit the maximum number of revoked users, but provides only t-conspiracy resistance.

In this algorithm the size of the predistributed data is $2 \log q$ and the size of the broadcast data is $(t + 1) \log q$. It seems to be more efficient than the revocation polynomial SKD and univariate polynomial secret sharing SKD. Unfortunately, efficiency cannot be easily compared between these algorithms, because the meaning of parameter t is different here. Here, it represents the maximum size of the authorized group G instead of the maximum number of revoked users. Thus, access polynomial SKD is suitable only to specific applications, where $|G| < |R|$. For more common applications, where $|G| >> |R|$, it is extremely expensive.

5) Summary: Polynomial based SKD algorithms are simple and efficient, especially in terms of communications overhead and computational cost. Conspiracy resistance and number of users which can be revoked is usually determined by the degree of applied polynomials. Polynomial calculations are linear and easily invertible. Thus, all polynomial based algorithms disclose some information about predistributed user data.

B. Exponential arithmetic based algorithms

Exponential arithmetic based SKD algorithms can be seen as an extension of polynomial based algorithms. They move polynomial computations to the exponents, to guarantee secrecy of the predistributed user data. The difficulty of computing a discrete logarithm in certain finite multiplicative groups, makes operations performed during key distribution very hard to invert.

Security of the algorithms relies on the difficulty of solving decisional Diffie-Hellman problem (DDH). We informally state DDH assumption here, referring the Reader to [24] for a more precise and detailed discussion. DDH is defined for any cyclic group H and generator g. The DDH assumption states that it is difficult to distinguish between the probability distributions of (g^a, g^b, g^{ab}) and (g^a, g^b, g^c), where a, b, and c are chosen randomly in $\{1, \ldots, |H|\}$. DDH is believed to be intractable in groups of large prime order.

First SKD algorithm based on exponential arithmetic was proposed by Staddon et al. in [3]. Later, Blundo et al. proposed an enhanced version of the algorithm in [5]. The key idea on which both algorithms are based is to do interpolation in the exponents. A concept of interpolation in the exponents was originally introduced in a secret sharing scheme by Feldman et al. [25], to enable each participant to verify his own share, received from a possibly dishonest dealer.

Since the algorithm of Blundo is simpler and more efficient than the algorithm of Staddon, we use it to illustrate the exponential arithmetic based approach. Let F_q^* be a multiplicative group of finite field of order q. Let F_p be a finite field of prime order p. Let g be a generator of a cyclic subgroup $H \subseteq F_q^*$ of prime order p in which DDH assumption holds. Let $I_U = \{x_i \in F_p\}_{U_i \in U}$ be a set of all indices assigned to the universe of users, where x_i is assigned to user U_i, let $I_R = \{x_i \in F_p\}_{U_i \in R}$ be the set of indices assigned to users which are revoked, and let $I_G = \{x_i \in F_p\}_{U_i \in G}$ be the set of indices assigned to authorized members of the group G. Given any polynomial $f(x) = a_t x^t + \ldots + a_1 x + a_0 \in H[x]$, let $g^{f(x)} = (g^{a_t}, \ldots, g^{a_0})$.

- *User U_i predistributed data:* $s_i = [x_i, s(x_i)]$, where $x_i \in I_U$ is a random index assigned to user U_i, and $s(x) \in F_p[x]$ is a random polynomial of degree t.
- *Broadcast public data:* $b = [g^v, z = g^{K+v \cdot s(0)}, \{(w_l, g^{v \cdot s(w_l)})\}_{w_l \in W}]$, where $W = \{w_1, \ldots, w_t\} \subseteq F_p$ is a set of distinct random indices such that $|W| = t$, $I_R \subseteq W$, $I_G \bigcap W = \emptyset$ and $0 \notin W$, and $v \in F_p$ is a random secret value generated by GM.
- *Session key calculation procedure for user U_i:*
  1) Evolve predistributed data by calculating $g^{v \cdot s(x_i)} = (g^v)^{s(x_i)}$.
  2) Recover $g^{v \cdot s(0)}$ by Lagrange interpolation in the exponents, based on t points $\{(w_l, g^{v \cdot s(w_l)})\}_{w_l \in W}$ received in b, and one point $(x_i, g^{v \cdot s(x_i)})$.
  3) Calculate session key $g^K = g^{K+v \cdot s(0)} / g^{v \cdot s(0)}$.

The scheme is very similar to the univariate polynomial secret sharing SKD, but all polynomial calculations are performed in exponents, and session key is defined as g^K. The session key is hidden by $g^{v \cdot s(0)}$. User U_i has to interpolate $g^{v \cdot s(x)}$, then calculate $g^{v \cdot s(0)}$ and obtain g^K. To correctly perform polynomial interpolation in exponents, U_i has to know at least $t+1$ distinct points of it. All users belonging to R have only access to t distinct points $\{(w_l, g^{v \cdot s(w_l)})\}_{w_l \in W}$, because $I_R \subseteq W$ and points which can be evolved from their predistributed data are already included in b. Thus, none of them is able to recover any information about session key from b. Even a collaborating group of such users is not able to recover any additional information about K. The algorithm allows to revoke up to t users, which is determined by the degree of polynomial $s(x)$. Predistributed data size is $2 \log p$ and broadcast data size is $(t+2) \log q + t \log p$.

Note that during the distribution of session key, only $g^{v \cdot s(x)}$ is made available to users $U_i \in G$. Providing that DDH assumption holds, values of $g^{v \cdot s(x)}$ cannot be used to obtain any information about $s(x)$. Thus, in this algorithm a single polynomial can be securely used in several sessions, if only v is chosen randomly and independently for each session. To be more precise, the algorithm is computationally secure, and the problem of obtaining values of $s(x)$ from $g^{v \cdot s(x)}$ can be reduced to DDH problem in H.

Most polynomial based algorithms can be transformed to exponential arithmetic in a similar way as algorithms of Staddon and Blundo. Efficiency of such exponential algorithms, in terms of broadcast data size, is slightly lower than the efficiency of the corresponding polynomial based algorithms, because they require a larger underlying group size in order to ensure that DDH problem is hard. Moreover, the computational cost of calculations in exponents is significantly higher than the cost of polynomial calculations. Nevertheless, these costs can be justified by the opportunity of reusing predistributed user data in several sessions.

C. Vector space secret sharing based algorithms

Vector space secret sharing introduced by Brickell et al. [26] can be seen as a generalization of the Shamir and Blakley secret sharing schemes. Recently it has been adopted in session key distribution algorithms. We first briefly introduce preliminaries of vector space secret sharing concept. Then, we focus on the usage of vector space secret sharing in session key distribution algorithms, and on the properties of those algorithms.

In secret sharing schemes, a secret value is shared among a set $U = \{U_1, \ldots, U_n\}$ of n players in such a way, that only qualified subsets of players can reconstruct the secret from their shares. The family of qualified subsets is called access structure. Access structure is a monotone increasing collection $\Gamma \subseteq 2^U \setminus \{\emptyset\}$ of non-empty subsets of U, that is, if $B \in \Gamma$ and $B \subseteq C \subseteq U$, then $C \in \Gamma$. The sets in Γ are called the authorized sets. A set B is called minimal set of Γ if $B \in \Gamma$, and for every $C \subset B$, $C \neq B$, it holds that $C \notin \Gamma$. The set of minimal authorized subsets of Γ is denoted by Γ_0 an is called the basis of Γ. Since Γ consists of all subsets of U that are supersets of a subset in the basis Γ_0, Γ is determined uniquely as a function of Γ_0. More formally, we have $\Gamma = \{C \subseteq U : B \subseteq C, B \in \Gamma_0\}$. The family of non-authorized subsets $\bar{\Gamma} = 2^U \setminus \Gamma$ is monotone decreasing, that is, if the following holds: $C \in \bar{\Gamma}$ and $B \subseteq C \subseteq U$, then $B \in \bar{\Gamma}$. The family of non-authorized subsets $\bar{\Gamma}$ is determined by the collection of maximal non-authorized subsets $\bar{\Gamma}_0$.

Vector space secret sharing scheme can be defined as follows. Let us suppose that a dealer is denoted by D and that there is public map $\psi : U \cup \{D\} \rightarrow (F_q)^l$, where q is a prime power and l is a positive integer. This map induces the monotone increasing access structure Γ, which satisfies the property: $B \in \Gamma$ if and only if vector $\psi(D)$ can be expressed as a linear combination of the vectors in the set

$\psi(B) = \{\psi(U_i)\}_{U_i \in B}$. An access structure Γ is said to be a vector space access structure, if it can be defined in the above way. To distribute a secret value $K \in F_q$, the dealer takes at random an element $v \in (F_q)^l$, such that $K = v \cdot \psi(D)$. The share of participant $U_i \in U$ is $s_i = v \cdot \psi(U_i)$. Let B be an authorized subset $B \in \Gamma$, then $\psi(D) = \sum_{U_k \in B} \lambda_k \psi(U_k)$, for some publicly known coefficients $\lambda_k \in F_q$. In order to recover secret K, participants of B pool their shares and compute:

$$\sum_{U_k \in B} \lambda_k s_k = v \cdot (\sum_{U_k \in B} \lambda_k \psi(U_k)) = v \cdot \psi(D) = K$$

On the other hand, one can show that if an unauthorized subset $C \notin \Gamma$ pool their shares, they can determine nothing about the value of K.

Vector space secret sharing was employed in several self-healing group key distribution schemes [10], [27]–[30], as a basis of the selective key distribution algorithm. SKD algorithm used in all these schemes is basically the same, and can be described in the following way. All operations take place in F_q, where q is a large prime number ($q > |U|$). Let ψ be some publicly known map $\psi : U \cup \{GM\} \rightarrow (F_q)^l$, satisfying the property $\psi(GM) \in \langle \psi(U_i) : U_i \in B \rangle \Leftrightarrow B \in \Gamma$, which defines vector space access structure Γ over the set U. The algorithm is generic, and as such, it does not specify details of the mapping function ψ.

- *User U_i predistributed data:* $s_i = [v \cdot \psi(U_i)]$, where $v \in (F_q)^l$ is a random secret vector,
- *Broadcast public data:* $b = [z = K + v \cdot \psi(GM), \{(U_k, v \cdot \psi(U_k))\}_{U_k \in W}]$, where $W \subseteq U$ is a random subset of users with minimal cardinality, such that $W \in \bar{\Gamma}_0$, $R \subseteq W$, and $G \cap W = \emptyset$.
- *Session key calculation procedure for user U_i:*
  1) Recover $v \cdot \psi(GM)$ using set $\{(U_k, v \cdot \psi(U_k))\}_{U_k \in W \cup \{U_i\}}$, by calculating $v \cdot \psi(GM) = \sum_{U_k \in W \cup \{U_i\}} \lambda_k v \cdot \psi(U_k)$, where $\lambda_k \in F_q$ are some publicly known coefficients. The calculation gives correct result, because $W \cup \{U_i\} \in \Gamma$.
  2) Calculate session key $K = z - v \cdot \psi(GM)$.

Vector space secret sharing based SKD algorithm provides some level of abstraction, since it does not impose any particular access structure. Although the vast majority of characteristics of the SKD algorithm, such as its efficiency and security, depend on the choice of ψ and consequently Γ, we decided to show some features which are inherent to the generic construction of the algorithm.

During the distribution of single key K, masking value $v \cdot \psi(GM)$ is disclosed to all users $U_i \in G$, so it can be securely used only in one session. It is also worth noting that, although the usage of general structures, instead of polynomial based structure, gives more flexible performance of the algorithm, in real applications the choice of access structure has to be made before system deployment, so the maximum number of users which can be revoked is actually predetermined and constrained by the chosen access structure Γ.

In [10], [27], [29], Shamir's (t, n)-threshold secret sharing scheme was proposed as an example realization of the generic access structure. It gives an algorithm identical to univariate polynomial secret sharing SKD algorithm. Furthermore, in [27] the author proposed a bipartite access structure, specialized for settings where an average number of users revoked in a single session is significantly lower than the maximum number of users, which can be revoked. To the best of our knowledge, there was no access structure proposed so far, which allows to achieve in standard settings a better performance that polynomial based SKD algorithms.

D. Bilinear pairings based algorithms

Recently, several SDK algorithms based on bilinear pairings have been proposed, including [31]–[33]. In these algorithms each user $U_i \in U$ is equipped with her own public/private key pair, as a part of her predistributed data. The public/private key pair is used to recover the session key from broadcast data, providing that U_i is an authorized group member. We can assume that broadcast data contains, in certain sense, ciphertexts obtained by encrypting session key K with a public key of each user. This is a great simplification, because actual algorithms are far more complicated, to achieve a better performance and security level. We do not present them in details in this paper, but instead we focus on their common properties. We refer the Reader to [31]–[33] for detailed descriptions.

The algorithms are computationally secure, since they are based on the asymmetric cryptography. They have a unique set of properties, solving most of the inherent problems of polynomial based algorithms. Predistributed user data can be used in several sessions, because private keys are not disclosed during session key distribution. Furthermore, any number of users can be revoked, and the algorithm can resist collusion of any coalition of non-authorized users. However, communication cost of the algorithms, in terms of broadcast data size, is prohibitively large. Broadcast data size depends on the total number of users, and is usually $O(|G|)$, which is not acceptable for large groups. Also, computational cost is higher than in polynomial based SKD algorithms, because the session key calculation involves pairing operations, which are more expensive than a scalar multiplication.

V. Predistributed secret data management

Selective key distribution algorithms specify how to distribute a single session key K to all authorized users $U_i \in G$, using broadcast media. Selective access to the broadcast session key is based on the fact that every user $U_i \in U$ is equipped with a different set of secret predistributed data, which is used during calculation of K. Life-cycle of the group key distribution schemes usually consists of a large number of sessions, and selective key distribution has to be repeated in each session. Therefore, every user has to store secret data needed for all expected sessions, which has to be delivered to users using a secure communication channel. This is usually done offline, before the deployment of the node, or when a new node joins the group.

Predistributed secret data management specifies how secret data stored by users is organized, and how it is used for selective key distribution. A structure of the predistributed data determines user storage requirements, and hence the

limitations of the scheme lifetime. There are two general policies used in predistributed secret data management in self-healing schemes:

1) independent secret data instances,
2) reusable secret data instances.

We present each approach and discuss its applicability to particular types of selective key distribution algorithms. We denote predistributed data of the user U_i by the sequence $S_i = [C_i, s_{i,1}, s_{i,2}, \ldots, s_{i,m}]$, where m is a maximum number of sessions, C_i is a configuration common to all sessions, and $s_{i,j}$ is an instance of the secret data used for selective key distribution in session j. Let $L(X)$ denote a size of the element X.

In the rest of the paper, we use following metrics to describe security of the key distribution schemes:

- *Forward secrecy strength* — maximum number of users revoked from group G in session j or before it, which colluding with each other are not able to recover any of subsequent session keys $(K_j, K_{j+1}, \ldots)$.
- *Backward secrecy strength* — maximum number of users who joined group G after session j, which colluding with each other are not able to recover any of past session keys $(K_1, \ldots, K_j)$.
- *Collusion resistance strength* — maximum number of users, belonging to the joined set of users revoked from group G in session l or before it, and users admitted to group G after session r, for any l, r such that $l < r$, which colluding with each other are not able to recover any of the session keys $(K_l, \ldots, K_r)$.

A. Independent secret data instances

A straightforward approach to extending selective key distribution algorithms to m sessions is by equipping users with m independent instances of secret data $s_{i,j}$, and by using a different instance in each session. This approach is the most universal, and it can be applied to any type of SKD algorithm. The instances $s_{i,j}$ are not related to each other, so even if some information about $s_{i,j}$ is disclosed during SKD in session j, it cannot be used to attack the scheme in subsequent sessions. Thus, schemes employing this technique achieve forward secrecy, with the strength determined by the conspiracy resistance of the SKD algorithm. Moreover, perfect backward secrecy can be ensured by equipping newly joined users with proper set of secret data instances. User U_i who joined the group in session j and is supposed to leave the group after session k, shall receive predistributed data $S_i = [C_i, s_{i,j}, \ldots, s_{i,k}]$.

Independent secret data instances are used in all secure self-healing schemes built on polynomial based SKD algorithms and vector space secret sharing based SKD algorithms, since all these algorithms disclose some information about the employed $s_{i,j}$.

This approach suffers from the high storage overhead. The size of the predistributed data grows linearly with m, since $L(S_i) = L(C_i) + mL(s_{i,j})$. Consequently, the maximum scheme lifetime is limited by the users' storage capabilities, and has to be predetermined before network deployment.

B. Reusable secret data instances

Storage overhead can be decreased by organizing predistributed user data in such a way, that there are some predefined relations between instances of secret data, or even the same secret data is used in several consecutive sessions. Security of such optimisations is strictly connected with the particular type of SKD algorithm applied in the scheme. Thus, we discuss them separately for each group of the algorithms.

1) Polynomial based algorithms and vector space secret sharing based algorithms: There have been several attempts to reuse predistributed secret data in self-healing schemes built on polynomial based SKD algorithms and vector space secret sharing based SKD algorithms, but all of them resulted in security vulnerabilities of the schemes. As far as we are concerned, it is not possible to do so without breaking scheme forward secrecy, since all these algorithms disclose some information about the employed $s_{i,j}$.

In a few schemes, including [14], [20], [22], authors suggested that the same masking polynomial can be used in all sessions, based on the assumption that the employed SKD algorithms are not leaking any information about $s_{i,j}$. This assumption is not correct, for the reasons explained in Section IV-A, and consequently all these schemes are insecure.

Dutta et al. in [34] significantly decreased the size of the predistributed user data, by using bivariate polynomial $\psi(x, y) \in F_q[x, y]$ of degree t in each dimension. The improvement is based on the assumption that polynomial $\psi(x, y)$ may be used to generate a sequence of random univariate polynomials. The scheme is built on revocation polynomial SKD algorithm. Every user $U_i \in U$ is equipped with predistributed data $S_i = [x_i, \psi(x_i, y)]$. In session j, GM calculates masking polynomial as $h_j(x) = \psi(x, \alpha_j)$, where α_j is some predefined constant, and U_i obtains her point on masking polynomial as $h_j(x_i) = \psi(x_i, \alpha_j)$. Unfortunately, the assumption, that polynomial $\psi(x, y)$ may be used to generate a sequence of random univariate polynomials, is incorrect and masking polynomials $h_j(x)$, used in consecutive sessions, are strictly related to each other. Since polynomial $h_j(x)$ is disclosed in session j to all users $U_i \in G_j$, every user who was an authorized group member for at least $t+1$ sessions, is able to recover the entire polynomial $\psi(x, y)$. This serious attack on the scheme forward secrecy was first uncovered in [35]. The same mechanism, based on the bivarate polynomial, was later used in several other schemes, namely [16], construction 3 of [9], [10], and [18]. It should be noted that all those schemes are only secure during the first t sessions.

2) Exponential arithmetic based algorithms: In self-healing schemes built on exponential arithmetic based algorithms, the same instance of predistributed secret data can be used in several sessions. In the extreme case, even a single instance may be used during the entire scheme lifetime. Therefore, exponential arithmetic based algorithms may be used to create so called *long-lived schemes*, that is schemes, whose lifetime does not have to be predetermined before deployment, because it can be easily extended during scheme operation.

In long-lived schemes proposed in [3], [5], every user $U_i \in G_0$ is fitted with predistributed data $S_i = [x_i, s_0(x_i), \ldots, s_{m-1}(x_i)]$. A sequence of secret values

$s_0(x_i), \ldots, s_{m-1}(x_i)$ is used repeatedly, thus a secret instance employed in session j is $s_{i,j} = s_{j \bmod m}(x_i)$. The size of predistributed data does not depend on the expected lifetime of the scheme, and in the extreme case, for the predefined parameter $m = 1$, the size is $L(S_i) = L(x_i) + L(s_0(x_i)) = 2 \log p$. Consequently, maximum scheme lifetime is not limited by user storage capabilities.

Schemes built on exponential arithmetic based algorithms achieve computational forward secrecy, of strength determined by the employed SKD algorithm. However, they do not provide backward secrecy. Let us assume, for simplicity, that $m = 1$. New user $U_i \notin G_{j-1}$ who joined group in session j, receives predistributed data $S_i = [x_i, s_0(x_i)]$. Since, the same instance of secret data is used in all sessions, U_i is able to recover session keys from collected broadcasts, for all sessions in which she was neither a member nor revoked. It means that U_i can recover session keys associated to previous groups to which she did not belong. This is a consequence of reusing secret data instances. Achieving backward secrecy in these schemes is an interesting open research problem.

3) Bilinear pairing based algorithms: In self-healing schemes built on bilinear pairing based algorithms the same instance of predistributed secret data is used during the entire scheme lifetime. They are long-lived schemes, since the maximum scheme lifetime is not limited by user storage capabilities.

These schemes, as opposed to the schemes built on exponential arithmetic based algorithms, achieve both forward and backward secrecy. It is ensured by the unique construction of the broadcast data, which explicitly specifies which users are entitled to obtain the session key in a particular session, whereas in exponential arithmetic based algorithms, broadcast data only specifies which users are revoked.

VI. Self healing mechanisms

The basic idea of self-healing technique is to add some redundant information to the broadcast message B, which would allow user nodes to recover previous session keys, lost due to communication errors. User nodes shall be able to recover lost session keys on their own, without any additional interaction with GM.

There are two versions of the requirements for the self-healing property. The first one was introduced by Staddon et al. [3], and according to it, lost session keys can be recovered by combining information from any message B_l preceding the lost packet B_j with the information from any message B_r following it. To be more formal: given any two session numbers l and r, such that $l < r$, there is an efficient algorithm ζ which for all $j : l < j < r$ can be used by node $U_i \in G_l \cap G_j \cap G_r$ to recover K_j knowing B_l and B_r, that is $K_j = \zeta(B_l, B_r, S_i)$. The second version of the requirement was proposed by Blundo et al. [5], as a simplification of the original one. According to it, lost session keys can be recovered from any single message B_r following the lost packet B_j. To be more formal: given any session number r, there is an efficient algorithm ζ which for all $j : j < r$ can be used by node $U_i \in G_j \cap G_r$ to recover K_j knowing B_r, that is $K_j = \zeta(B_r, S_i)$.

The self-healing mechanism specifies how broadcast messages B_k are constructed and how users can use redundant data from them, to recover lost session keys. There are two main classes of the self-healing techniques applied in self-healing schemes, namely:

1) independent session keys self-healing,
2) related session keys self-healing.

We present each self-healing technique and discuss its main characteristics. Let B_k be a message broadcast by GM in session k, and let b_k be broadcast data, needed for selective distribution of session key K_k. Let $L(X)$ denote the size of the element X.

A. Independent session keys self-healing

Independent session keys self-healing techniques assume that there are no relations between session keys in different sessions, so every key has to be distributed separately by applying an appropriate SKD algorithm. Consequently, message B_k broadcast in session k must contain b_k to allow distribution of the key K_k, and some partial information about b_l from other sessions $l \neq k$ to allow recovery of lost keys.

This class of self-healing techniques provides three types of recovery of lost session keys, namely:

1) key recovery from sandwiching broadcasts,
2) key recovery from any pair of broadcasts,
3) key recovery from single broadcast.

First self healing schemes proposed in [3], [11] employed self healing techniques providing key recovery from sandwiching broadcasts. They assume that an authorized user shall be able to recover lost session key only if she has received at least one key distribution broadcast preceding the lost packet and at least one key distribution broadcast following it. In other words, in order to recover a lost session key, the user must have received key distribution broadcasts for any two sessions which "sandwich" the session corresponding to the lost key distribution broadcast. Later, this requirement was slightly relaxed to achieve better efficiency of the schemes. Several techniques providing key recovery from any pair of broadcasts were proposed. They allow the authorized user to recover lost session key, by combining information from any key distribution broadcast following the lost packet with the information from any other key distribution broadcast. Finally, the requirement was simplified by Blundo et al. [5], and a few techniques providing key recovery from a single broadcast were proposed. In these techniques the user can recover all lost session keys (for sessions in which she belongs to the group) by using single key distribution broadcast from any later session.

We discuss in detail each type of independent session keys self-healing techniques, by describing most significant techniques and pointing out their main characteristics.

1) Key recovery from sandwiching broadcasts: In self healing techniques providing key recovery from sandwiching broadcasts, lost session keys can be recovered by combining information from any message B_l preceding the lost packet B_j with information from any message B_r following it. To be more formal: given any two session numbers l and r, such that $l < r$, there is an efficient algorithm ζ which for any

$j : l < j < r$, can be used by node $U_i \in G_l \cap G_j \cap G_r$ to recover K_j knowing B_l and B_r, that is $K_j = \zeta(B_l, B_r, S_i)$.

Staddon et al. in [3] proposed first self-healing technique with this property. Later, Liu et al. in [11] employed a similar technique in his self-healing key distribution scheme. In both techniques, GM randomly splits each session key K_i into two shares K_i^1 and K_i^2. Then, in session k, GM broadcasts message B_k of the following form:

$$B_k = \left[b_1^1, \ldots, b_{k-1}^1, b_k, b_{k+1}^2, \ldots, b_m^2\right]$$

where, b_i^1 is a data used by SKD algorithm to selectively distribute key share K_i^1, b_i^2 is a data used by SKD algorithm to selectively distribute key share K_i^2, and b_i is a data used by SKD algorithm to selectively distribute session key K_i. Authorized users are able to calculate from B_k current session key K_k, shares of previous keys $(K_1^1, \ldots, K_{k-1}^1$ and shares of future keys $(K_{k+1}^2, \ldots, K_m^2)$. To recover lost session key K_j, user U_i calculates share K_j^2 from some previous broadcast, and share K_j^1 from some later broadcast, and then, she calculates $K_j = K_j^1 + K_j^2$.

We illustrate this approach on self-healing technique proposed in [11], since it is based on revocation polynomial SKD algorithm, which is simpler than SKD algorithm used in [3], and as such it is easier to understand. In this technique, GM randomly splits each session key K_i into two t-degree polynomials, $p_i(x)$ and $q_i(x)$, such that $K_i = p_i(x)+q_i(x)$. The GM then distributes in separate broadcasts points on polynomial $p_i(x)$ and points on polynomial $q_i(x)$. The authorized user, who correctly received points on both polynomials, is able to recover lost K_i.

- *Message broadcast by GM in session k:* $B_k = [I_{R_1}, \ldots, I_{R_k}, P_1(x), \ldots, P_k(x), Q_k(x), \ldots, Q_m(x)]$, where, $P_i(x) = r_i(x)p_i(x) + h_i(x)$ and $Q_i(x) = q_i(x) + f_i(x)$ are polynomials broadcast to selectively distribute key shares $p_i(x)$ and $q_i(x)$, according to revocation polynomial SKD algorithm.
- *Lost key K_j recovery procedure for user U_i, who received any pair of broadcasts B_l and B_r, such that $l < j < r$ and $U_i \in G_l \cap G_j \cap G_r$:*
  1) Recover $p_j(x_i)$ from $P_j(x)$, received in message B_r, using predistributed data S_i,
  2) Recover $q_i(x_i)$ from $Q_i(x)$, received in message B_l, using predistributed data S_i,
  3) Calculate session key $K_j = p_j(x_i) + q_j(x_i)$.

From a single broadcast B_k, authorized user U_i is able to recover key shares $(p_1(x_i), \ldots, p_k(x_i))$ and $(q_k(x_i), \ldots, q_m(x_i))$. Hence, she has only one pair of shares, namely $p_k(x_i)$ and $q_k(x_i)$, which can be used to recover current session key K_k. Single B_k does not give any information about other session keys. Recovered shares of future keys should be stored by U_i, since they may be combined with key shares from later broadcasts to recover lost session keys.

This technique provides self healing mechanism, satisfying strict requirement that lost session key may only be recovered from sandwiching broadcasts. Unfortunately, the technique is very expensive in terms of the broadcast size, since $L(B_k) = mL(b_k)$. Moreover, every user has to store recovered shares of all future keys, which may introduce significant storage overhead.

Blundo et al. [5] proved that the requirement, that lost session key may only be recovered from sandwiching broadcasts, does not increase security of the self healing scheme. Since user U_i, who correctly received broadcast B_k is able to obtain K_k if only she belongs to authorized group G_k, she should also be able to recover K_k if B_k was lost. The only condition should be that U_i must be a member of G_k to be able to recover the lost session key K_k using self-healing mechanism. There are no security gains from introducing any additional constraints on the self-healing mechanism. Therefore, several self-healing techniques providing key recovery from any pair of broadcasts and key recovery from a single broadcast were proposed.

2) Key recovery from any pair of broadcasts: In self healing techniques providing key recovery from any pair of broadcasts, lost session keys can be recovered by combining information from any message B_r following the lost packet B_j with information from any other message B_l. To be more formal: given any two session numbers l and r, such that $l < r$, there is an efficient algorithm ζ which for any $j : j < r$, can be used by node $U_i \in G_l \cap G_j \cap G_r$ to recover K_j knowing B_l and B_r, that is $K_j = \zeta(B_l, B_r, S_i)$.

This property is achieved by including in every broadcast message B_k, data chunks $(b_1, \ldots, b_{k-1})$, used by SKD algorithm to selectively distribute previous session keys, mixed with each other in such a way, that at least two different messages B_{k_1}, B_{k_2} are necessary to obtain value of the single chunk. It can be realized in many ways, but two forms of broadcast message B_k are most common in the existing self healing schemes:

1) $B_k = [b_1 + b_2, \ldots, b_1 + b_{k-1}, b_k]$
2) $B_k = [b_1 + b_2, \ldots, b_{k-2} + b_{k-1}, b_k]$

The first one was introduced by Blundo et al. in construction 2 of [5]. The second one was proposed by Hong et al. in construction 1 of [12], and was later employed in several schemes, including construction 1 of [22], [23], and construction 1 of [16]. We briefly describe techniques of Blundo and Hong, to show how this approach can be applied for different SKD algorithms.

Construction 2 of [5] is based on univariate polynomial secret sharing SKD algorithm. The self-healing technique can be described as follows.

- *Message broadcast by GM in session k:* $B_k = [z_1 + z_2, \ldots, z_1 + z_{k-1}, z_k, (\{(w_l, s_i(w_l))\}_{w_l \in W_i})_{i=1,\ldots,k}]$, where $z_i = s_i(0) + K_i$ and $\{(w_l, s_i(w_l))\}_{w_l \in W_i}$ are values broadcast to selectively distribute key K_i, according to univariate polynomial secret sharing SKD algorithm.
- *Lost key K_j recovery procedure for user U_i, who received any pair of broadcasts B_l and B_r, such that $j < r$, $l < r$ and $U_i \in G_l \cap G_j \cap G_r$:*
  1) Obtain z_1 by subtracting the value z_l given in B_l, from the value $z_1 + z_l$ given in B_r, that is $z_1 = (z_1 + z_l) - z_l$,
  2) Calculate z_j by subtracting z_1 from the value z_1+z_j given in B_r, that is $z_j = (z_1 + z_j) - z_1$,
  3) Recover session key K_j from z_j and

$\{(w_l, s_j(w_l))\}_{w_l \in W_j}$, using predistributed data S_i.

Only value z_k, used in distribution of current session key K_k, is broadcast in the plain form. All previous $(z_2, \ldots, z_{k-1})$ are masked by addition of value z_1, so at least one another broadcast message is necessary to obtain their values. Once the lost z_j is recovered, calculation of the lost session key K_j proceeds exactly the same way as during ordinary selective key distribution.

Construction 1 of [12] is based on revocation polynomial SKD algorithm. Self-healing technique proposed in this construction can be described as follows.

- *Message broadcast by GM in session k:* $B_k = [I_{R_1}, \ldots, I_{R_k}, P_1(x) + P_2(x), \ldots, P_{k-2}(x) + P_{k-1}(x), P_k(x)]$, where set of indices I_{R_i} and polynomial $P_i(x) = r_i(x) \cdot K_i + h_i(x)$ is a data broadcast to selectively distribute session key K_i, according to revocation polynomial SKD algorithm.
- *Lost key K_j recovery procedure for user U_i, who received any pair of broadcasts B_l and B_r, such that $j < r$, $l < r$ and $U_i \in G_l \cap G_j \cap G_r$:*
  1) Obtain polynomial $P_{l-1}(x)$ by subtracting polynomial $P_l(x)$ given in B_l from polynomial $P_{l-1}(x) + P_l(x)$ given in B_r, that is $P_{l-1}(x) = (P_{l-1}(x) + P_l(x)) - P_l(x)$.
  2) Repeat operation from step 1 to obtain sequence of polynomials $(P_{l-1}(x), P_{l-2}(x), \ldots, P_j(x))$.
  3) Recover session key K_j from $P_j(x)$ and I_{R_j}, using predistributed data S_i.

In this technique, similarly to the technique of Blundo, only current session key K_k can be recovered from B_k, since only polynomial $P_k(x)$ is distributed in plain form. All previous polynomials $(P_1(x), \ldots, P_{k-1}(x))$ are masked by summation of adjacent polynomials, so at least one another broadcast message is necessary to obtain their values. Once the lost $P_j(x)$ is recovered, calculation of the lost session key K_j proceeds according to revocation polynomial SKD algorithm.

Both techniques have basically the same characteristics. They are more efficient in terms of broadcast size than techniques providing key recovery from sandwiching broadcasts, since $L(B_k) = kL(b_k)$, but they still introduce significant communication overhead. Every user has to store the latest data chunk b_i, to allow recovery of lost future keys.

3) Key recovery from a single broadcast: In self healing techniques providing key recovery from a single broadcast, lost session keys can be recovered from any single message B_r following the lost packet B_j. To be more formal: given any session number r, there is an efficient algorithm ζ, which for all $j : j < r$ can be used by node $U_i \in G_j \cap G_r$ to recover K_j knowing B_r, that is $K_j = \zeta(B_r, S_i)$.

This technique was first proposed by Blundo et al. in construction 3 of [5]. Self-healing is achieved by including in every broadcast message B_k, all previously broadcast data chunks $(b_1, \ldots, b_{k-1})$, used by SKD algorithm to selectively distribute previous session keys. In other words, self-healing is realized by retransmitting in each session all data broadcast in previous sessions, that is $B_i = [B_{i-1}, b_i]$. In session k, GM broadcasts message B_k of the following form:

$$B_k = [b_1, \ldots, b_k]$$

Recovery of lost session key K_j from any following broadcast B_k, proceeds exactly the same way as selective key distribution of K_j, since B_k contains the same data chunk b_j which was originally broadcast in lost message B_j.

Construction 3 of [5] is based on univariate polynomial secret sharing SKD algorithm. The self-healing technique employed in this scheme can be described as follows.

- *Message broadcast by GM in session k:* $B_k = \left[z_1, \ldots, z_k, (\{(w_l, s_i(w_l))\}_{w_l \in W_i})_{i=1,\ldots,k}\right]$, where $z_i = s_i(0) + K_i$ and $\{(w_l, s_i(w_l))\}_{w_l \in W_i}$ are values broadcast to selectively distribute key K_i, according to univariate polynomial secret sharing SKD algorithm.
- *Lost key K_j recovery procedure for user U_i, who received any broadcast B_r, such that $j < r$, and $U_i \in G_j$:*
  1) Recover session key K_j from z_j and $\{(w_l, s_j(w_l))\}_{w_l \in W_j}$, using predistributed data S_i, according to univariate polynomial secret sharing SKD algorithm.

This technique is the simplest and the most universal self-healing technique, which can be used with any type of SKD algorithm. It was employed in many self-healing schemes, including construction 4 of [5], construction 2 of [12], [18]–[20], [32], [33] and construction 2 of [16], [22]. Communication overhead introduced by the self-healing mechnism is the same as for techniques providing key recovery from any pair of broadcasts—size of the single broadcast message is $L(B_k) = kL(b_k)$. However, users do not need to store any data received in previous sessions, so it can be considered as truly stateless technique.

4) Summary: Independent session keys self-healing techniques provide self-healing mechanisms which does not affect security of the self-healing scheme. Backward and forward secrecy, as well as collusion resistance of the scheme is ensured by usage of SKD algorithm for distribution of any session key or key share. Scheme security depends only on employed selective key distribution algorithm and applied policy of the predistributed secret data management.

Unfortunately, all independent session keys self-healing techniques suffer from a significant communication overhead. There have been several attempts to reduce the size of the broadcast message. The most useful approach was proposed in [36] and construction 4 of [11]. It allows for a tradeoff between communication overhead and self-healing capability, by introduction of the *sliding window* δ, so that only redundant information for the sessions that fall into this window is broadcast. Broadcast message B_k contains data needed for distribution of the current session key K_k, and data used for recovery of the previous and the future δ session keys. Consequently, self-healing property is limited to only last δ session keys. This limitation may be justified by the fact that in majority of applications users are only interested in the several most recent session keys and old session keys are worthless. We illustrate a sliding window approach on the technique providing key recovery from a single broadcast but it can be easily applied to any other independent session keys self-healing technique. In the technique providing key recovery

from a single broadcast, with sliding window δ, broadcast message B_k is of the form $B_k = [b_{k-\delta}, \ldots, b_k]$. Note that this slight modification changes size of the broadcast message from $O(k)$ to $O(\delta)$, which, depending on the choice of parameter δ, may be a significant improvement.

B. Related session keys self-healing

In related session keys self-healing techniques session keys are mathematically related to each other. The relation is publicly known, and can be used by authorized users to calculate lost session keys, so only current session key has to be distributed by applying SKD algorithm. Message B_k broadcast in session k must include b_k to allow distribution of the key K_k, but does not need to contain b_l from other sessions $l \neq k$. Thus, broadcast message can be significantly smaller than in the case of independent session keys self-healing techniques.

The relation between session keys shall be constructed in such a way, that it would not be possible for unauthorized users to gain any information about session keys they are not entitled to. To be more specific, self-healing technique should achieve: forward secrecy, backward secrecy and collusion resistance. This is usually done by employing one-way hash functions. A hash function takes a binary string of an arbitrary length as an input, and outputs a binary string of a fixed length. A one-way function ϑ satisfies the following two properties:

1) given x, it is easy to compute y such that $y = \vartheta(x)$,
2) given y, it is computationally infeasible to compute x such that $y = \vartheta(x)$.

The one-way hash function, denoted in the rest of this section by H, is a hash function with one-way property.

1) Keyed permutation: Dutta et al. in [14], [15] introduced self-healing technique based on keyed permutation. In this technique session keys are calculated by applying keyed permutation $f_{\beta_i} : F_q \rightarrow F_q$, that is $K_i = f_{\beta_i}(K_{i-1})$. All operations take place in a finite field F_q, where q is a large prime number. Permutation family is publicly known, but permutation keys $\beta_1, \ldots, \beta_m$ are randomly chosen by GM and distributed only to authorized users.

Self-healing technique proposed by Dutta uses symmetric key encryption to realize keyed permutation over F_q. Let $E_x(\cdot)$, $D_x(\cdot)$ be respectively an encryption and corresponding decryption function using key x.

- *Relation between session keys:* $K_i = E_{\beta_i}(K_{i-1})$, where $K_0 \in F_q$ is a secret prime key randomly selected by GM, and $\beta_i \in F_q$ is a permutation key randomly selected by GM in session i.
- *Message broadcast by GM in session k:* $B_k = [b_{K_k}, E_{K_k}(\beta_1), \ldots, E_{K_k}(\beta_k)]$, where b_{K_k} is a data chunk used by applied SKD algorithm to selectively distribute K_k.
- *Lost key K_j recovery procedure for user U_i, who received any pair of broadcasts B_l and B_r, such that $l < j < r$ and $U_i \in G_l \cap G_r$:*
  1) Recover session key K_r from b_{K_r} given in B_r, using predistributed data S_i, according to employed SKD algorithm.
  2) Obtain sequence of permutation keys $\beta_{l+1}, \ldots, \beta_j$ by decrypting cipher strings $E_{K_r}(\beta_{l+1}), \ldots, E_{K_r}(\beta_j)$ given in B_r, using session key K_r.
  3) Recover session key K_l from b_{K_l} given in B_l, using predistributed data S_i, according to employed SKD algorithm.
  4) Recover sequence of session keys $K_{l+1}, \ldots, K_j$ by calculating $K_i = E_{\beta_i}(K_{i-1})$ for each $i = l+1, \ldots, j$.

Self-healing is achieved by distribution, in each session k, permutation keys $\beta_1, \ldots, \beta_k$, defining relations between all the previous session keys. Users knowing some past session key K_l and permutation keys $\beta_{l+1}, \ldots, \beta_{k-1}$, are able to recover all subsequent session keys $K_{l+1}, \ldots, K_{k-1}$, by calculating $K_{l+1} = E_{\beta_{l+1}}(K_l), K_{l+2} = E_{\beta_{l+2}}(K_{l+1}), \ldots, K_{k-1} = E_{\beta_{k-1}}(K_{k-2})$. Selective access to broadcast permutation keys is ensured by symmetric encryption using current session key K_k. Since permutation keys $\beta_1, \ldots, \beta_k$ are broadcast in the encrypted form $E_{K_k}(\beta_1), \ldots, E_{K_k}(\beta_k)$, only users knowing K_k are able to recover them. Consequently, access to the permutation keys is restricted to users $U_i \in G_k$ by SKD algorithm, employed to distribute current session key.

Technique of Dutta is based on the assumption that keyed permutation, realized by symmetric encryption $E_{\beta_i}(\cdot)$, is a one way function. Unfortunately, the assumption is incorrect since knowing key β_i it is easy to invert encryption function $E_{\beta_i}(\cdot)$ by corresponding decryption function $D_{\beta_i}(\cdot)$. It was shown in [35] that the technique of Dutta does not provide backward secrecy. A user who joined the group at some session $k > 1$ and correctly recovered session key K_k, and sequence of permutation keys $\beta_1, \ldots, \beta_k$, is able to calculate all previous session keys $K_{k-1} = D_{\beta_k}(K_k), K_{k-2} = D_{\beta_{k-1}}(K_{k-1}), \ldots, K_1 = D_{\beta_2}(K_2)$.

The problem with backward secrecy was recently solved by Gu et al. [30]. The proposed technique is basically the same as the technique of Dutta, but keyed permutation is realized by cryptographically secure one-way hash function $H(\cdot)$. Relation between session keys is defined by equation $K_i = H(\beta_i, K_{i-1})$.

The technique provides computational forward and backward secrecy, but cannot resist collusion between revoked and newly joined users, and as such it does not allow a user to rejoin. Given any two users U_1 and U_2, where $U_1 \in R_l$ is a user who was revoked in session l, $U_2 \in J_r$ is a user who joined in session r, and $l < r$, the users collaborating with each other are able to recover session keys $K_l, \ldots, K_{r-1}$, that they are not entitled to. This is possible, because U_1 knows some previous session keys $K_i : i < l$, and U_2 can recover sequence of permutation keys $\beta_1, \ldots, \beta_r$. Hence, by sharing data with each other, they are able to calculate session keys, using lost key recovery procedure.

Self-healing technique based on the keyed permutation was originally applied to revocation polynomial SKD algorithm, but it is universal and can be used with any other secure selective key distribution algorithm. Furthermore, it is more efficient, in terms of the broadcast size, than independent session keys self-healing techniques, since data chunks generated by SKD algorithm are usually bigger than ciphertext created

by encryption of permutation keys. In this technique, size of the single broadcast message is $L(B_k) = L(b_k) + k \log q$.

2) One-way hash chains: Self-healing technique based on hash chains was first introduced by Jiang et al. in key distribution scheme providing only implicit node revocation [4]. Later, a similar technique was adopted by Dutta et al. for scheme supporting dynamic node revocation [8]. Technique of Dutta has become one of the most popular self-healing methods applied in the self-healing key distribution schemes, because of its simplicity and efficiency. It was employed, with some improvements, in several self-healing schemes, including [9], [10], [17], [21], [28], [29]. We first introduce the concept of one-way hash chains, and then we describe technique of Dutta, to discuss the main characteristics of self-healing methods based on the one-way hash chains.

A one-way hash chain is a sequence of hash values $(x_0, x_1, \ldots, x_n)$, where x_0 is a chain seed, and next values are calculated by repeatedly applying on it the same one-way hash function $H : F_q \rightarrow F_q$, that is $(x_1 = H(x_0), x_2 = H(x_1), \ldots, x_n = H(x_{n-1}))$. Given any element $x_j : 0 < j < n$, it is computationally infeasible to calculate any x_l, such that $0 \leq l < j$, but it is easy to obtain any element x_r, such that $j < r \leq n$, by calculating $x_r = H^{r-j}(x_j)$, where $H^i(\cdot)$ means that function H is applied i times.

Self healing technique proposed in [8], is based on two one way hash chains: forward chain, denoted by $(K_0^F, K_1^F, \ldots, K_m^F)$, and backward chain, denoted by $(K_0^B, K_1^B, \ldots, K_m^B)$. Forward hash chain is derived by GM from randomly selected forward seed $K_0^F \in F_q$, by calculating $(K_1^F = H(K_0^F), \ldots, K_m^F = H^m(K_0^F))$. Similarly, backward hash chain is derived from randomly selected backward seed $K_0^B \in F_q$ by calculating $(K_1^B = H(K_0^B), \ldots, K_m^B = H^m(K_0^B))$. The self-healing technique can be described as follows.

- *Relation between session keys:* $K_i = K_i^F + K_{m-i+1}^B$,
- *Predistributed data of user U_i, who joined the group in session l:* $S_i = [K_l^F, s_{i,l}, \ldots, s_{i,m}]$, where, $s_{i,j}$ is a predistributed secret data used by SKD algorithm in session j.
- *Message broadcast by GM in session k:* $B_k = [b_{K_{m-k+1}^B}]$, where, $b_{K_{m-k+1}^B}$ is a data chunk used by applied SKD algorithm to selectively distribute hash value K_{m-k+1}^B.
- *Lost key K_j recovery procedure for user U_i, who received any broadcast B_r, such that $j < r$ and $U_i \in G_j \cap G_r$:*
  1) Recover hash value K_{m-r+1}^B from $b_{K_{m-r+1}^B}$ given in B_r, using predistributed data S_i, according to employed SKD algorithm.
  2) Obtain hash value K_{m-j+1}^B, by calculating $K_{m-j+1}^B = H^{r-j}(K_{m-r+1}^B)$.
  3) Derive hash value K_j^F from K_l^F given in predistributed data, by calculating $K_j^F = H^{j-l}(K_l^F)$.
  4) Calculate session key $K_j = K_j^F + K_{m-j+1}^B$.

The technique achieves computational backward and forward secrecy. Backward secrecy is ensured by forward hash chain. User U_i, who joined the group in session l, is equipped with hash value K_l^F, so it is able to calculate sequence $(K_l^F, \ldots, K_m^F)$, but it is computationally infeasible for her to calculate any of previous hash values $(K_1^F, \ldots, K_{l-1}^F)$. Thus, user U_i does not have hash values necessary to recover any session key K_j distributed before she joined the group ($j < l$). Forward secrecy, in turn, is achieved by selective distribution of hash values from backward hash chain. In session k, hash value K_{m-k+1}^B is distributed with use of SKD algorithm, so only authorized users $U_i \in G_k$ are able to recover K_{m-k+1}^B, and calculate sequence $(K_{m-k+2}^B, \ldots, K_m^B)$, needed for recovery of lost session keys. Because of the one-way property of hash function H, it is computationally infeasible to calculate from K_{m-k+1}^B any of hash values $(K_1^B, \ldots, K_{m-k}^B)$, which are necessary to recover any information about future session keys $(K_{k+1}, \ldots, K_m)$.

This self-healing method is very efficient in terms of broadcast size, since it does not require any additional information, beside data chunk needed for selective distribution of current session key, to be broadcast to provide recovery of lost session keys. Also, user storage costs are very low, because user needs to store only single element of forward hash chain to be able to recover lost session keys. However, maximum number of sessions m has to be constant and predetermined before system deployment, because entire backward hash chain has to be derived by GM during the initialization. Thus, it cannot be applied in long-lived schemes. Moreover, the computational cost of evaluating one-way hash function is significantly higher than the cost of polynomial calculations, so hash values should be precomputed if possible, to decrease number of hash function evaluations, at the cost of additional storage space.

Unfortunately, the technique of Dutta is vulnerable to *collusion attack*, and as such it does not allow for user rejoin. Given any two users U_1 and U_2, where $U_1 \in R_l$ is a user who was revoked in session l, $U_2 \in J_r$ is a user who joined in session r, and $l < r$, if the users collaborate with each other, they are able to recover session keys $K_l, \ldots, K_{r-1}$, that they are not entitled to. This is because U_1 is still able to calculate any forward hash values $(K_l^F, \ldots, K_m^F)$ even after revocation, and U_2 is able to calculate all backward hash values $(K_1^B, \ldots, K_{m-r}^B)$ upon recovering K_{m-r+1}^B.

Lack of *collusion resistance* is an inherent issue of the self-healing techniques based on the one-way hash chains. There have been several attempts to solve this problem, including [9], [17], [21], [28], but according to our knowledge, it is not possible to do so without sacrificing desirable properties of the self-healing technique. We shortly describe the most important proposals, discussing various approaches to achieve collusion resistance.

Tian et al. claim that the self-healing technique based on one-way hash chains, proposed in [28] resists collusion between the newly joined users and the revoked users. This is achieved by predetermining the expected life cycle of the users before they join the group and by providing them with minimal subset of random numbers $\alpha_1, \ldots, \alpha_m$ generated by GM, which are used during calculation of session keys. Every user U_i with assigned life cycle (l, r), meaning that she joins the group in session l and is expected to leave it in session $r + 1$, is equipped with a set of secret predistributed data $S_i = [K_l^F, \alpha_l, \ldots, \alpha_r, s_{i,l}, \ldots, s_{i,r}]$, where $s_{i,j}$ is secret data used by the SKD algorithm in session j. In session $r + 1$ life cycle of the user U_i expires and she is implicitly revoked,

since she cannot participate in session key distribution any longer. Recovery of lost session key K_j proceeds similarly to [8] but the key is calculated as $K_j = K_j^F + (K_{m-j+1}^B \oplus \alpha_j)$.

Since knowledge of random number α_j is necessary to calculate K_j, the technique resists collusion between the newly joined users and the revoked users whose life cycle expired. Given any two users U_1 and U_2, where $U_1 \in R_l$ is a user who was revoked in session l, which was after the last session of her life cycle, and $U_2 \in J_r$ is a user who joined in session r, and $l < r$, the users are not able to recover any of the session keys $K_l, \ldots, K_{r-1}$, even collaborating with each other, because none of them knows anything about values $\alpha_l, \ldots, \alpha_{r-1}$. However, newly joined users collaborating with revoked users, whose life cycle has not expired, are still able to recover session keys they are not entitled to. Therefore, this technique solves only a special case of collusion attacks, where users are implicitly revoked by expiration of their life cycle, and in other cases it just reduces the number of session keys that collaborating users can recover. Note that in most applications it is very hard to accurately predetermine a user life cycle, as the main purpose of using session key distribution scheme is to dynamically revoke and add users, during system operation. Moreover, in this technique, user storage overhead is increased in comparison to the [8], because every user has to store a subset of values $\alpha_1, \ldots, \alpha_m$, determined by his life-cycle.

Dutta et al. [9] proposed self-healing technique based on one-way hash chains, which can be seen as a simplification of the self-healing technique of Tian. It eliminates forward hash chain, since it does not contribute to the security of the self-healing method when random values $\alpha_1, \ldots, \alpha_m$ are used, and uses only backward hash chain. Every user U_i with the assigned life cycle (l, r), is equipped with a set of secret predistributed data $S_i = [\alpha_l, \ldots, \alpha_r, s_{i,l}, \ldots, s_{i,r}]$, where $s_{i,j}$ is secret data used by the SKD algorithm in session j. During the recovery of lost session key K_j, hash value K_{m-j+1}^B is recovered the same way as in [8], and then the key is calculated as $K_j = \alpha_j + K_{m-j+1}^B$. This technique achieves the same level of collusion resistance as [28], that is it resists collusion between newly joined users and revoked users whose life cycle expired, but it is more efficient in terms of computational cost, because all calculations performed on the forward hash chain are eliminated.

To improve collusion resistance, authors of [9] introduced an additional assumption, that users cannot be revoked before their life cycle is over. In other words, only implicit user revocation, taking place during expiration of the user life cycle, is allowed. In our opinion, the assumption limits usefulness of this self-healing technique, since usually it is very hard to accurately predetermine a user life cycle, as the main purpose of using session key distribution scheme is to dynamically revoke and add users, during system operation. Also, there is no point in using any SKD algorithm in such a scheme, since user life cycle is fully predetermined and selective access to the distributed session keys can be easily ensured by providing users with the minimal subsets of random values $\alpha_1, \ldots, \alpha_m$, determined by their life cycles.

Authors of [17] claim that they fully solved collusion problem in self-healing technique based on one-way hash chains. The technique is based on two one way hash chains: forward chain $(K_0^F, K_1^F, \ldots, K_m^F)$, and backward chain $(K_0^B, K_1^B, \ldots, K_m^B)$. During the initialization GM precomputes both hash chains, and it also generates a sequence of random values $\alpha_1, \ldots, \alpha_m$. Then, every user U_i with the assigned life cycle (l, r), is equipped with a set of secret predistributed data $S_i = [K_l^F, K_{m-r+1}^B, \alpha_l, \ldots, \alpha_r, s_{i,l}, \ldots, s_{i,r}]$, where $s_{i,j}$ is secret data used by the SKD algorithm in session j. Message broadcast in session k is of the form $B_k = [b_{c_k}, c_k\alpha_1(c_1 + c_2), c_k\alpha_2(c_2 + c_3), \ldots, c_k\alpha_{k-2}(c_{k-2} + c_{k-1}), c_k\alpha_{k-1}c_{k-1}]$, where $c_i \in F_q$ is a random number corresponding to session i, and b_{c_k} is a data chunk used by applied SKD algorithm to selectively distribute value c_k. Lost session key K_j is recovered by user U_i from broadcast B_r, such that $j < r$, by first recovering c_r, according to applied SKD algorithm, then computing sequence of values $c_{r-1}, c_{r-2}, \ldots, c_j$, deriving hash values K_j^F, K_{m-j+1}^B and finally calculating session key $K_j = (K_j^F + K_{m-j+1}^B)c_j$.

Since knowledge of random number c_j is necessary to calculate K_j, and U_i must possess the entire sequence of predistributed random values $\alpha_j, \ldots, \alpha_{r-1}$ to be able to recover c_j from B_r, the technique resists collusion between the newly joined users and revoked users, if only life cycles of the newly joined users do not overlap with the life cycles of the revoked users. That is, given any two users U_1 and U_2, where $U_1 \in R_l$ is a user who was revoked in session l, but according to the assigned life cycle she was originally expected to leave group in session $k+1$, $l \leq k$, and $U_2 \in J_r$ is a user who joined in session r, and $l \leq k < r$, the users are not able to recover any of session keys $K_l, \ldots, K_{r-1}$, even collaborating with each other, because none of them know anything about values $c_{k+1}, \ldots, c_{r-1}$. However, newly joined users collaborating with the revoked users, whose life cycles overlap, are still able to recover session keys they are not entitled to. Moreover, the technique is much more complicated and significantly less efficient in terms of the broadcast size, than techniques in [9], [28] - size of the broadcast message transmitted in session k is $O(k)$. Therefore, this technique improves collusion resistance at the cost of communication overhead, but it does not fully solve collusion problem in self-healing methods based on the one-way hash chains. This is because introduction of the additional sequence of random values $c_1, \ldots, c_m$ used during session key calculation, which are common for all users, by definition cannot fully solve collusion issues.

3) Summary: Related session keys self-healing techniques provide self-healing mechanisms which are very efficient in terms of broadcast size, but they affect security of the self-healing session key distribution scheme. They ensure computational backward and forward secrecy, but they cannot achieve full collusion resistance of the scheme without losing its efficiency.

C. Drawbacks of self-healing mechanisms

Most of the self-healing techniques add some redundant information to the broadcast message B, to allow user nodes to recover previous session keys, lost due to communication errors. This introduces communicational overhead, which highly

depends on the employed solution. Only techniques based on one-way hash chains can provide self-healing without any additional communicational overhead.

Since self-healing mechanism introduces extra cost, it should only be used in networks with an unreliable channel, where packet losses are expected. For optimum channels without communication errors, were retransmissions are not needed, self-healing mechanism would only add extra-cost without any benefits. However, in case of unreliable channels, addition of redundant information to B is more efficient in terms of communicational cost, than performing retransmissions on packet losses. Message retransmissions should be avoided, if possible, since they are costly and require feedback connection from receiver nodes to GM. In large networks retransmissions can overload GM.

VII. Evaluation of self-healing schemes

In this section we compare several representative group key distribution schemes with self healing property. According to our knowledge, this is the first such comprehensive comparison available in literature.

A. Evaluation criteria

There is a wide number of group key distribution schemes with self-healing property in the literature, so we decided to develop a set of common evaluation criteria to facilitate comparison of these schemes. Existing solutions usually present some tradeoff between scheme performance and security level. Our metrics allow for the evaluation of scheme performance in terms of communication, computational and storage overhead, and address selected security aspects. We organized these metrics into three main categories: performance, survivability and security.

Performance metrics:

- $L(B_j)$ — size of the message broadcast by GM in session j.
- $L(S_i^j)$ — amount of storage space occupied by secret predistributed data S_i, stored at user U_i, who joined the group in session j and is not expected to leave it before the last session of the scheme.
- $L(data_{GM})$ — amount of storage space occupied by all scheme data stored at GM.
- C_{B_j} — computational complexity of calculation performed by GM to create broadcast message B_j.
- $C_{K_j}(B_j, S_i)$ — computational complexity of calculation performed by authorized user U_i to recover session key K_j from broadcast B_j, using her secret data S_i.
- $C_{K_j}(B_l, B_r, S_i)$ — computational complexity of calculation performed by authorized user U_i to recover lost session key K_j from two broadcasts $\{B_l, B_r\}_{l<j<r}$ or from single broadcast B_r, using her secret data S_i.

Survivability metrics:

- $|G|_{max}$ — group size limit.
- *Sessions limit* — maximum number of sessions a scheme can work without reinitialization.
- *Revocation limit* — maximum cumulative number of users, which can be revoked during the entire lifetime of the scheme.
- *Self-healing window* — maximum number of previous session keys $(K_1, \ldots, K_{j-1})$ which can be recovered by self-healing mechanism, using data from current broadcast B_j (and optionally data from sandwiching broadcast, depending on the type of applied self-healing technique).
- *Late join* — scheme property allowing to dynamically add new users to the group during scheme operation (in later sessions).
- *User rejoin* — scheme property allowing to add users to the group, who were previously revoked, without disclosing to them session keys which were distributed during the revocation period.
- *Dynamic revocation* — scheme property allowing to dynamically remove users from the group at any session (user life time does not need to be predetermined during join operation).

Security metrics:

- *Forward secrecy strength* — maximum number of users revoked from group G up to and including session j, which colluding with each other are not able to recover any of the subsequent session keys $(K_j, K_{j+1}, \ldots)$.
- *Backward secrecy strength* — maximum number of users who joined group G after session j, which colluding with each other are not able to recover any of the past session keys $(K_1, \ldots, K_j)$.
- *Collusion resistance strength* — maximum number of users, belonging to the joined set of users revoked from group G up to and including session l, and users admitted to the group G after session r, for any l, r such that $l < r$, which colluding with each other are not able to recover any of the session keys $(K_l, \ldots, K_r)$.
- *Session key security* — type of session key security ensured by the scheme (i.e. unconditional or computational).

B. Selected schemes

Each self-healing scheme can be seen as a combination of basic elements, which we have identified and described in Section III, that is: selective key distribution mechanism, predistributed secret data management, and self-healing mechanism. There is a wide number of solutions presented in literature, which represent almost every possible permutation of the basic elements. Fig. 3 presents a classification of the most important schemes, based on the applied selective key distribution mechanism and self-healing mechanism. In case of papers introducing multiple schemes, we use 'Cx' to denote particular construction, for example 'C3 of [3]' denotes the construction 3 proposed in paper [3].

We picked up a sample of eight schemes from various papers, representing different approaches, for detailed evaluation. Selected constructions can be described as follows:

C3 of [3] Bivariate polynomial secret sharing SKD algorithm, independent secret data instances, independent session keys self-healing technique, key recovery from sandwiching broadcasts.

C4 of [11] Revocation polynomial SKD algorithm, independent secret data instances, independent ses-

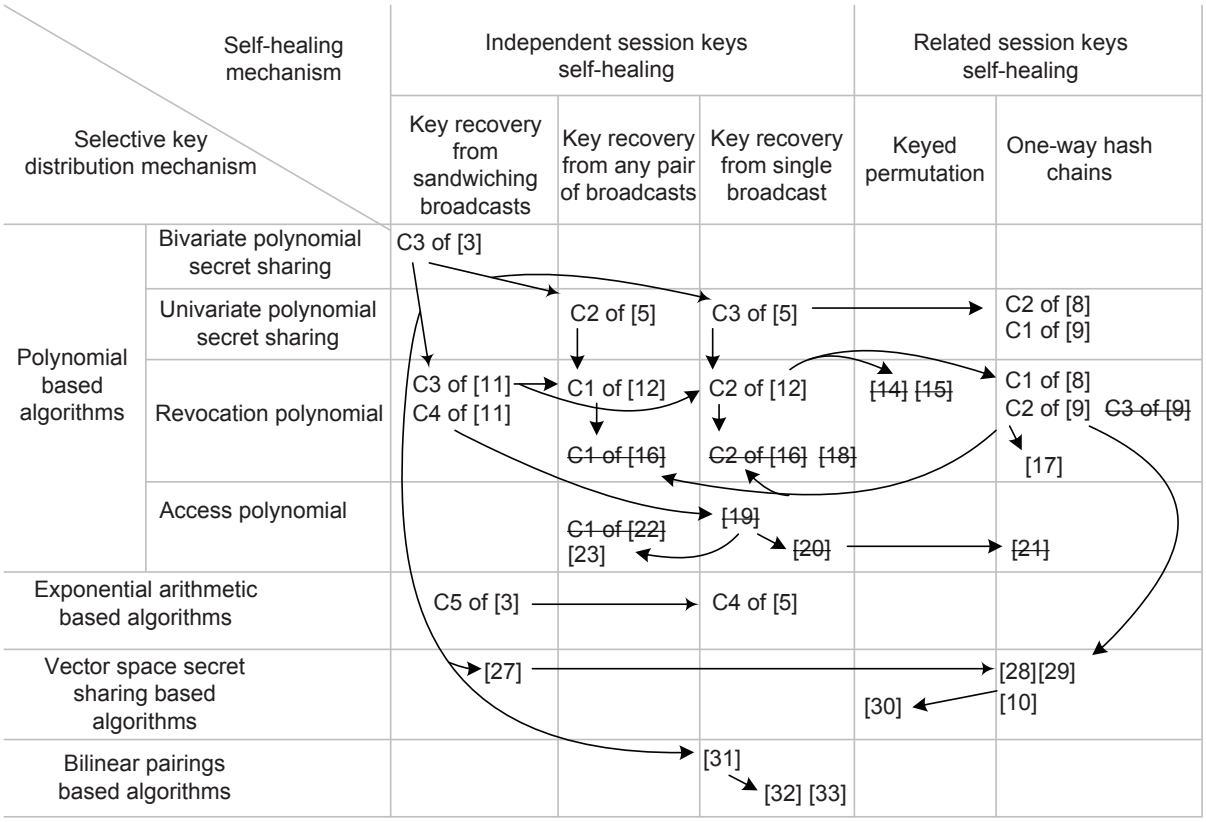

Fig. 3. Two-dimensional classification of the self-healing group key distribution schemes, based on the applied selective key distribution mechanism and self-healing mechanism. Relations between different solutions are illustrated as a direct acyclic graph, where nodes represent proposed schemes, and arrows indicate that successor is based on the predecessor and is supposed to improve it (we use similar representation as [1]). Schemes which were proved to be insecure are marked with a strikethrough line.

sion keys self-healing technique, key recovery from sandwiching broadcasts, sliding window.

C3 of [5] Univariate polynomial secret sharing SKD algorithm, independent secret data instances, independent session keys self-healing technique, key recovery from single broadcast.

C2 of [12] Revocation polynomial SKD algorithm, independent secret data instances, independent session keys self-healing technique, key recovery from single broadcast.

[23] Access polynomial SKD algorithm, independent secret data instances, independent session keys self-healing technique, key recovery from any pair of broadcasts.

C1 of [8] Revocation polynomial SKD algorithm, independent secret data instances, related session keys self-healing technique, based on one-way hash chains.

C4 of [5] Exponential arithmetic based SKD algorithm, reusable secret data instances, independent session keys self-healing technique, key recovery from single broadcast.

[32] Bilinear pairings based SKD algorithm, reusable secret data instances, independent session keys self-healing technique, key recovery from single broadcast.

C. Comparison of the selected schemes

Algorithms used in self-healing solutions are deterministic, so scheme efficiency and complexity can be evaluated for a given set of parameters (such as m, t, q), without any network simulations. Thus, in the comparison we provide precise analytical formulas instead of the numerical data.

Comparison of the selected self-healing schemes, based on the subset of the metrics proposed in Section VII-A, is presented in tables I and II. Performance metrics are provided in table I, survivability and security metrics are included in table II. Efficiency of each construction is described by communicational cost, determined by the size of a single broadcast message ($L(B_j)$), and by storage requirements for user node ($L(S_i^j)$). Computational complexity of each scheme is assessed by estimating time complexity of two basic operations performed by a user node: key calculation ($C_{K_j}(B_j, S_i)$), and lost key recovery through self-healing ($C_{K_j}(B_l, B_r, S_i)$). Time complexity is described asymptotically (using big-O notation), to facilitate comparison of schemes scalability. Due to the variety of available algorithms, parameter $M(X)$ is used to denote complexity of multiplication in algebraic structure X and parameter $E(H_1, H_2)$ is used to denote complexity of bilinear pairing $H_1 \times H_1 \rightarrow H_2$. Since GM is assumed to be a resource rich node, her storage requirements ($L(data_{GM})$) and complexity metrics (C_{B_j}) are not provided in the comparison.

TABLE I
COMPARISON OF SELF HEALING SCHEMES (PART 1)

Scheme	Parameters	Metrics			
		$L(B_j)$	$L(S_i^j)$	$C_{K_j}(B_j, S_i)$	$C_{K_j}(B_l, B_r, S_i)$
C3 of [3]	$m \in N$, $t \in N$, F_q – finite field of order q, $M(F_q)$ – complexity of multiplication in F_q	$\left(mt^2 + 3mt + m\right) \log q$	$\left((m-j+1)^2 + 1\right) \log q$	$O\left(t^2 \cdot M(F_q)\right)$	$O\left(t^2 \cdot M(F_q)\right)$
C4 of [11]	$m \in N$, $t \in N$, $\delta \in N$, F_q – finite field of order q, $M(F_q)$ – complexity of multiplication in F_q	$\left(\delta(3t+2) + \sum_{i=j-\delta+1}^{j} \lvert R_i \rvert\right) \log q$	$2(m-j+1) \log q$	$O\left(t \cdot M(F_q)\right)$	$O\left(t \cdot M(F_q)\right)$
C3 of [5]	$m \in N$, $t \in N$, F_q – finite field of order q, $M(F_q)$ – complexity of multiplication in F_q	$(2tj + j) \log q$	$(m-j+1) \log q$	$O\left(t^2 \cdot M(F_q)\right)$	$O\left(t^2 \cdot M(F_q)\right)$
C2 of [12]	$m \in N$, $t \in N$, F_q – finite field of order q, $M(F_q)$ – complexity of multiplication in F_q	$\left(j(t+1) + \sum_{i=1}^{j} \lvert R_i \rvert\right) \log q$	$(m-j+1) \log q$	$O\left(t \cdot M(F_q)\right)$	$O\left(t \cdot M(F_q)\right)$
[23]	$m \in N$, $t \in N$, F_q – finite field of order q, $M(F_q)$ – complexity of multiplication in F_q	$(t+1)(j-1) \log q$	$2(m-j+1) \log q$	$O\left(t \cdot M(F_q)\right)$	$O\left(t(j-l) \cdot M(F_q)\right)$
C1 of [8]	$m \in N$, $t \in N$, F_q – finite field of order q, $M(F_q)$ – complexity of multiplication in F_q	$(\lvert R_j \rvert + t + 1) \log q$	$(m-j+2) \log q$	$O\left(t \cdot M(F_q)\right)$	$O\left((t+r-l) \cdot M(F_q)\right)$
C4 of [5]	$m \in N$, $t \in N$, F_p – finite field of order p, $H \subseteq F_q^\star$ – cyclic subgroup of order p, $M(F_p)$ – complexity of multiplication in F_p, $M(F_q^\star)$ – complexity of multiplication in $F_q^\star$	$(2j + jt) \log q + jt \log p$	$m \log p$	$O\left(t \log p \cdot M(F_q^\star) + t^2 \cdot M(F_p)\right)$	$O\left(t \log p \cdot M(F_q^\star) + t^2 \cdot M(F_p)\right)$
[32]	H_1 – cyclic additive group of prime order q, of points on an elliptic curve over F_p, H_2 – cyclic multiplicative group of prime order q, $L(H_1)$ – size of the element of H_1, $M(H_1)$ – complexity of multiplication in H_1, $E(H_1, H_2)$ – complexity of bilinear pairing $H_1 \times H_1 \to H_2$	$\sum_{i=1}^{j} (\lvert G_i \rvert \cdot L(H_1) + \log q)$	$2 \cdot L(H_1)$	$O\left(\lvert G_j \rvert^3 \cdot M(H_1) + E(H_1, H_2)\right)$	$O\left(\lvert G_j \rvert^3 \cdot M(H_1) + E(H_1, H_2)\right)$

TABLE II
COMPARISON OF SELF HEALING SCHEMES (PART 2)

Scheme	Metrics										
	$\lvert G \rvert_{max}$	Sessions limit	Revocation limit	Self-healing window	Late join	User rejoin	Dynamic revocation	Forward secrecy strength	Backward secrecy strength	Collusion resistance strength	Session key security
C3 of [3]	$\lvert U \rvert$	m	t	$j-1$	Yes	Yes	Yes	t	$\lvert U \rvert$	t	unconditional
C4 of [11]	$\lvert U \rvert$	m	t	$\delta - 1$	Yes	Yes	Yes	t	$\lvert U \rvert$	t	unconditional
C3 of [5]	$\lvert U \rvert$	m	t	$j-1$	Yes	Yes	Yes	t	$\lvert U \rvert$	t	unconditional
C2 of [12]	$\lvert U \rvert$	m	t	$j-1$	Yes	Yes	Yes	t	$\lvert U \rvert$	t	unconditional
[23]	$t-1$	m	$\lvert U \rvert$	$j-1$	Yes	Yes	Yes	t	$\lvert U \rvert$	t	unconditional
C1 of [8]	$\lvert U \rvert$	m	t	$j-1$	Yes	No	Yes	t	$\lvert U \rvert$	1	computational, inverting one-way hash function
C4 of [5]	$\lvert U \rvert$	unlimited	t	$j-1$	Yes	No	Yes	t	0	0	computational, DDH assumption
[32]	$\lvert U \rvert$	unlimited	$\lvert U \rvert$	$j-1$	Yes	Yes	Yes	$\lvert U \rvert$	$\lvert U \rvert$	$\lvert U \rvert$	computational, BDH assumption

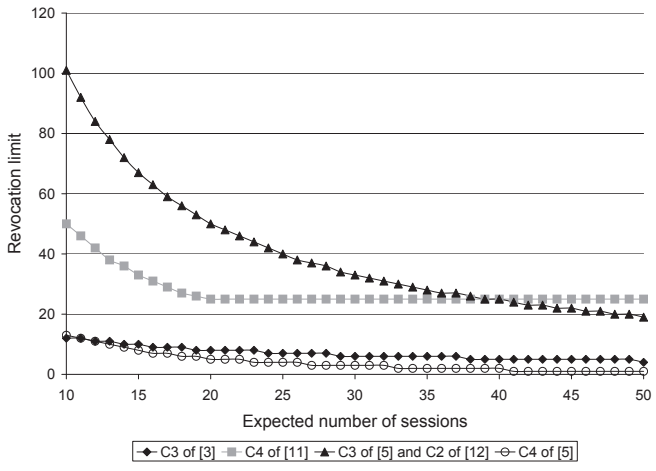

Fig. 4. Revocation limits that can be achieved in the selected polynomial based schemes and exponential arithmetic based schemes, for the expected number of sessions varying from 10 to 50, and size of the broadcast message limited by 64 KB.

To illustrate communicational cost of the selected schemes, we assess schemes' capabilities in a network, in which size of the broadcast packet is limited by 64 KB. The assessment is based on the metrics provided in table I. We assume that q defining a finite field F_q in the polynomial based schemes (i.e. C3 of [3], C4 of [11], C3 of [5], C2 of [12], [23], C1 of [8]) is a 128-bit integer. For scheme C4 of [5] based on the exponential arithmetic, we assume that p and q defining a cyclic subgroup $H \subseteq F_q^\star$, are 160-bit integer and 1520-bit integer respectively. For scheme [32] based on the bilinear pairing, we assume that both p and q are 160-bit integers. Moreover, we assume that self-healing window δ used in the scheme C4 of [11], is equal to 20. We refer the Reader to [37] for a detailed discussion of the secure key sizes in various algebraic structures.

Fig. 4 presents revocation limits that can be achieved in the selected schemes, for the expected number of sessions varying from 10 to 50, and size of the broadcast message limited by 64 KB. Construction C1 of [8] was not shown in this figure, because its self-healing technique is based on the one-way hash chains, and the size of its broadcast message does not

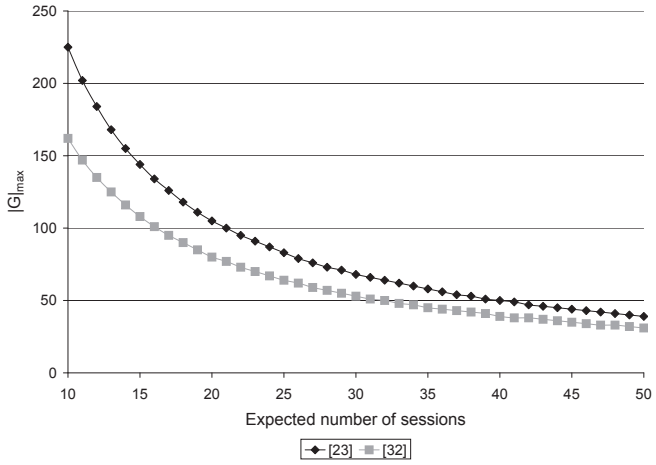

Fig. 5. Maximum size of the group ($|G|_{max}$) that can be achieved in constructions [23] and [32], for the expected number of sessions varying from 10 to 50, and size of the broadcast message limited by 64 KB.

depend on the expected number of sessions. Revocation limit achieved by this scheme is equal to 1023 users for the given size of the broadcast packet. The scheme based on the one-way hash chains is much more efficient than schemes using independent session keys self-healing techniques. However, they can be significantly improved using a sliding window approach, which can be observed for C4 of [11].

In constructions [23] and [32] any number of users can be revoked, but the maximum size of the group $|G|_{max}$ is limited by the size of broadcast message. Fig. 5 presents group size limits that can be achieved in these constructions, for the expected number of sessions varying from 10 to 50, and the size of the broadcast message limited by 64 KB. It can be seen that these constructions should only be used for small groups.

VIII. Conclusions

This article presented a survey on self-healing group key distribution schemes. A survey and analysis by the authors have shown that all existing schemes can be decomposed into three elements: *selective key distribution mechanism*, *predistributed secret data management* and *self-healing mechanism*. Techniques applied in each element were grouped into several categories. Each category was discussed by introducing example solutions. This way of categorizing the available schemes gives one the opportunity to establish a deeper insight into available self-healing key distribution approaches and to discuss the strengths and weaknesses of each one.

Polynomial based algorithms are the most popular selective key distribution mechanisms, but the efficiency of schemes built on these algorithms could not be substantially (and securely) improved. In our opinion, exponential arithmetic based algorithms are the most promising ones, since they do not disclose any information about predistributed secret data, and as such, they allow to reuse secret data instances. They are significantly more efficient than bilinear pairings based algorithms, which also possess this property. However, the problem of achieving backward secrecy in schemes built on exponential arithmetic based algorithms has to be addressed to enable wide use of these algorithms in self-healing key distribution schemes.

Techniques based on one-way hash chains are the most efficient self-healing mechanisms, but they cannot achieve full collusion resistance without losing their efficiency. Thus, in applications with strict security requirements, it would be more practical to use one of the independent session keys self-healing techniques, combined with sliding window approach.

Self-healing property ensures that the schemes are resilient to packet loss, but it usually introduces communicational overhead and some revocation latency. Thus, it should only be used in networks with an unreliable channel, where packet losses are expected. For optimum channels without communication errors, self-healing mechanism would only add extra-cost without any benefits. However, in case of unreliable channels, addition of redundant information to broadcast message is more efficient in terms of the communicational cost, than performing retransmissions on packet losses.

Further research is needed to develop secure and efficient long-lived self-healing group key distribution scheme. In most existing solutions, maximum scheme lifetime is limited and has to be predetermined before system deployment. There are two factors restricting lifetime of the scheme: maximum number of sessions, limited by the structure of users predistributed secret data, and maximum number of users which can be revoked, limited by employed SKD algorithm. In most applications, it may be difficult to accurately determine a desirable number of sessions and a number of users that can be revoked. On the other hand, these parameters should not be overestimated, since they usually have a strict impact on the scheme efficiency.

Schemes using bilinear pairings based algorithms are long-lived, but they are not feasible because of the prohibitively large communicational overhead. In our opinion, exponential arithmetic based algorithms are the most promising approach to create efficient long-lived schemes. Existing solutions using exponential arithmetic based algorithms can support an unlimited number of sessions. However, future research needs to be conducted to ensure backward secrecy in this class of schemes, and to extend the maximum number of users which can be revoked. In our opinion, it is possible to make significant improvements by using *stateful* algorithms in future schemes.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable feedback which helped to improve the paper.

References

[1] B. Tian, S. Han, S. Parvin, J. Hu, and S. Das, "Self-healing key distribution schemes for wireless networks: A survey," *The Computer Journal*, vol. 54, no. 4, pp. 549–569, 2011. [Online]. Available: http://comjnl.oxfordjournals.org/content/54/4/549.abstract

[2] Q. Wang, "Practicality analysis of the self-healing group key distribution schemes for resource-constricted wireless sensor networks," *Communications and Mobile Computing, International Conference on*, vol. 0, pp. 37–40, 2011.

[3] J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin, and D. Dean, "Self-healing key distribution with revocation," *Security and Privacy, IEEE Symposium on*, vol. 0, p. 241, 2002.

[4] Y. Jiang, C. Lin, M. Shi, and X. S. Shen, "Self-healing group key distribution with time-limited node revocation for wireless sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 14 – 23, 2007, security Issues in Sensor and Ad Hoc Networks. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870506000448

[5] C. Blundo, P. D'arco, A. De Santis, and M. Listo, "Design of self-healing key distribution schemes," *Des. Codes Cryptography*, vol. 32, pp. 15–44, May 2004. [Online]. Available: http://dx.doi.org/10.1023/B:DESI.0000029210.20690.3f

[6] M. Naor and B. Pinkas, "Efficient trace and revoke schemes," in *Proc. 4th International Conference on Financial Cryptography*, ser. FC '00. London, UK, UK: Springer-Verlag, 2001, pp. 1–20. [Online]. Available: http://portal.acm.org/citation.cfm?id=647504.728500

[7] J. Anzai, N. Matsuzaki, and T. Matsumoto, "A quick group key distribution scheme with entity revocation," in *Proc. International Conference on the Theory and Applications of Cryptology and Information Security: Advances in Cryptology*, ser. ASIACRYPT '99. London, UK: Springer-Verlag, 1999, pp. 333–347. [Online]. Available: http://portal.acm.org/citation.cfm?id=647095.716850

[8] R. Dutta, E.-C. Chang, and S. Mukhopadhyay, "Efficient self-healing key distribution with revocation for wireless sensor networks using one way key chains," in *Proc. 5th international conference on Applied Cryptography and Network Security*, ser. ACNS '07. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 385–400. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-72738-5_25

[9] R. Dutta, S. Mukhopadhyay, and T. Dowling, "Trade-off between collusion resistance and user life cycle in self-healing key distributions with t-revocation," in *Applications of Digital Information and Web Technologies, 2009. ICADIWT '09. Second International Conference on the*, aug. 2009, pp. 603 –608.

[10] R. Dutta, S. Mukhopadhyay, and M. Collier, "Computationally secure self-healing key distribution with revocation in wireless ad hoc networks," *Ad Hoc Networks*, vol. 8, no. 6, pp. 597 – 613, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870509001176

[11] D. Liu, P. Ning, and K. Sun, "Efficient self-healing group key distribution with revocation capability," in *Proc. 10th ACM conference on Computer and communications security*, ser. CCS '03. New York, NY, USA: ACM, 2003, pp. 231–240. [Online]. Available: http://doi.acm.org/10.1145/948109.948141

[12] D. Hong and J.-S. Kang, "An efficient key distribution scheme with self-healing property," *IEEE Commun. Lett.*, vol. 9, no. 8, pp. 759 – 761, aug 2005.

[13] C. Blundo, P. DArco, and A. De Santis, "Definitions and bounds for self-healing key distribution schemes," in *Automata, Languages and Programming*, ser. Lecture Notes in Computer Science, J. Daz, J. Karhumaki, A. Lepisto, and D. Sannella, Eds. Springer Berlin / Heidelberg, 2004, vol. 3142, pp. 179–194, 10.1007/978-3-540-27836-8_22. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-27836-8_22

[14] R. Dutta, Y. D. Wu, and S. Mukhopadhyay, "Constant storage self-healing key distribution with revocation in wireless sensor network," in *Communications, 2007. ICC '07. IEEE International Conference on*, june 2007, pp. 1323 –1328.

[15] R. Dutta, S. Mukhopadhyay, and S. Emmanuel, "Low bandwidth self-healing key distribution for broadcast encryption," in *Modeling Simulation, 2008. AICMS 08. Second Asia International Conference on*, may 2008, pp. 867 –872.

[16] T. Yuan, J. Ma, Y. Zhong, and S. Zhang, "Self-healing key distribution with revocation and collusion resistance for wireless sensor networks," in *Proc. 2008 International Multi-symposiums on Computer and Computational Sciences*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 83–90. [Online]. Available: http://portal.acm.org/citation.cfm?id=1491268.1493180

[17] H. Z. ChunLai Du, MingZeng Hu and W. Zhang, "Anti-collusive self-healing key distribution scheme with revocation capability," *Information Technology Journal*, vol. 8, pp. 619–624, 2009. [Online]. Available: http://scialert.net/abstract/?doi=itj.2009.619.624

[18] S. Han, B. Tian, M. He, and E. Chang, "Efficient threshold self-healing key distribution with sponsorization for infrastructureless wireless networks," *Trans. Wireless. Comm.*, vol. 8, pp. 1876–1887, April 2009. [Online]. Available: http://dx.doi.org/10.1109/TWC.2009.080046

[19] X. Zou and Y.-S. Dai, "A robust and stateless self-healing group key management scheme," in *Communication Technology, 2006. ICCT '06. International Conference on*, nov. 2006, pp. 1 –4.

[20] B. Tian, S. Han, and T. Dillon, "An efficient self-healing key distribution scheme," in *New Technologies, Mobility and Security, 2008. NTMS '08.*, nov. 2008, pp. 1 –5.

[21] R. Dutta, S. Mukhopadhyay, and T. Dowling, "Enhanced access polynomial based self-healing key distribution," in *Security in Emerging Wireless Communication and Networking Systems*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, O. Akan, P. Bellavista, J. Cao, F. Dressler, D. Ferrari, M. Gerla, H. Kobayashi, S. Palazzo, S. Sahni, X. S. Shen, M. Stan, J. Xiaohua, A. Zomaya, G. Coulson, Q. Gu, W. Zang, and M. Yu, Eds. Springer Berlin Heidelberg, 2010, vol. 42, pp. 13–24, 10.1007/978-3-642-11526-4_2. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-11526-4_2

[22] T. Yuan, J. Ma, Y. Zhong, and S. Zhang, "Efficient self-healing key distribution with limited group membership for communication-constrained networks," in *Proc. 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing - Volume 01*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 453–458. [Online]. Available: http://portal.acm.org/citation.cfm?id=1488726.1489026

[23] ——, "Self-healing key distribution with limited group membership property," in *Proc. 2008 First International Conference on Intelligent Networks and Intelligent Systems*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 309–312. [Online]. Available: http://portal.acm.org/citation.cfm?id=1471609.1472959

[24] D. Boneh, "The decision diffie-hellman problem," in *Proceedings of the Third International Symposium on Algorithmic Number Theory*. London, UK: Springer-Verlag, 1998, pp. 48–63. [Online]. Available: http://portal.acm.org/citation.cfm?id=648184.749735

[25] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *Proc. 28th Annual Symposium on Foundations of Computer Science*, ser. SFCS '87. Washington, DC, USA: IEEE Computer Society, 1987, pp. 427–438. [Online]. Available: http://dx.doi.org/10.1109/SFCS.1987.4

[26] E. F. Brickell, "Some ideal secret sharing schemes," in *Proceedings of the workshop on the theory and application of cryptographic techniques on Advances in cryptology*. New York, NY, USA: Springer-Verlag New York, Inc., 1990, pp. 468–475. [Online]. Available: http://portal.acm.org/citation.cfm?id=111563.111607

[27] G. Saez, "On threshold self-healing key distribution schemes," in *Cryptography and Coding*, ser. Lecture Notes in Computer Science, N. Smart, Ed. Springer Berlin / Heidelberg, 2005, vol. 3796, pp. 340–354, 10.1007/11586821_23. [Online]. Available: http://dx.doi.org/10.1007/11586821_23

[28] B. Tian, S. Han, T. S. Dillon, and S. Das, "A self-healing key distribution scheme based on vector space secret sharing and one way hash chains," in *Proc. 2008 International Symposium on a World of Wireless, Mobile and Multimedia Networks*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 1–6. [Online]. Available: http://portal.acm.org/citation.cfm?id=1549824.1550157

[29] R. Dutta, S. Mukhopadhyay, A. Das, and S. Emmanuel, "Generalized self-healing key distribution using vector space access structure," in *Proc. 7th international IFIP-TC6 networking conference on AdHoc and sensor networks, wireless networks, next generation internet*, ser. NETWORKING'08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 612–623. [Online]. Available: http://portal.acm.org/citation.cfm?id=1792514.1792582

[30] J. Gu and Z. Xue, "An efficient self-healing key distribution with resistance to the collusion attack for wireless sensor networks," in *Communications (ICC), 2010 IEEE International Conference on*, may 2010, pp. 1 –5.

[31] X. Du, Y. Wang, J. Ge, and Y. Wang, "An id-based broadcast encryption scheme for key distribution," *IEEE Trans. Broadcast.*, vol. 51, no. 2, pp. 264 – 266, june 2005.

[32] B. Tian, S. Han, and T. Dillon, "A self-healing and mutual-healing key distribution scheme using bilinear pairings for wireless networks," in *Embedded and Ubiquitous Computing, 2008. EUC '08. IEEE/IFIP International Conference on*, vol. 2, dec. 2008, pp. 208 –215.

[33] S. Han, B. Tian, Y. Zhang, and J. Hu, "An efficient self-healing key distribution scheme with constant-size personal keys for wireless sensor networks," in *Communications (ICC), 2010 IEEE International Conference on*, may 2010, pp. 1 –5.

[34] R. Dutta and S. Mukhopadhyay, "Improved self-healing key distribution with revocation in wireless sensor network," in *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE*, march 2007, pp. 2963 –2968.

[35] V. Daza, J. Herranz, and G. Sáez, "Flaws in some self-healing key distribution schemes with revocation," *Inf. Process. Lett.*, vol. 109, pp. 523–526, May 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1519558.1519948

[36] S. M. More, M. Malkin, J. Staddon, and D. Balfanz, "Sliding-window self-healing key distribution," in *Proc. 2003 ACM workshop on*

Survivable and self-regenerative systems: in association with 10th ACM Conference on Computer and Communications Security, ser. SSRS '03. New York, NY, USA: ACM, 2003, pp. 82–90. [Online]. Available: http://doi.acm.org/10.1145/1036921.1036930

[37] A. K. Lenstra and E. R. Verheul, "Selecting cryptographic key sizes." *Journal of Cryptology*, vol. 14, pp. 255–293, 2001.

Tomasz Rams received the M.Sc. degree in Electronics and Telecommunication from AGH University of Science and Technology, Krakow, Poland in 2008. He worked for Motorola Poland as a software engineer (2006–2011). Since 2011, he is a senior R&D engineer at Nokia Siemens Networks R&D Center in Krakow. Tomasz Rams is currently working toward the Ph.D. degree in Telecommunications at the AGH University of Science and Technology. His research interests include key management protocols, secure group communication, security of machine-to-machine systems and wireless ad hoc networks.

Piotr Pacyna, Ph.D, is a researcher and lecturer with the Department of Telecommunications at AGH University of Science and Technology in Krakow, Poland. His research focuses on next-generation IP networks, mobility management and security. Piotr Pacyna spent sabbatical leaves in CNET France Telecom in France and in Universidad Carlos III de Madrid in Spain. He has been active FP5, FP6 and FP7 ACTS and IST integrated research projects. Piotr Pacyna co-authored several research papers published in conferences and in renown journals, and contributed to ITU-T SG17.