Journal of Information Assurance and Security.
ISSN 1554-1010 Volume 14 (2019) pp. 048-066

Security Analysis of Proxy Cryptography Based Group Key Management Schemes for Dynamic and Wireless Networks Under Active Outsider Attack Model

Shravani Mahesh Patil[1] **and Purushothama B R**[2]

[1]National Institute of Technology Goa, Farmagudi, Ponda, Goa, 403401
patilmshravani@gmail.com

[2]National Institute of Technology Goa, Farmagudi, Ponda, Goa, 403401
puru@nitgoa.ac.in

***Abstract*:** **The existing group key management schemes have considered passive adversary model to show that schemes satisfy forward and backward secrecy requirements. A more pragmatic model is the active outsider adversary model wherein the adversary compromises the valid group member. The practicality of application of most of the schemes is hampered due to their insecurity under the active adversary model. In this paper, we analyze the group key management schemes based on proxy cryptography for their security under active outsider adversary model. For these schemes to be secure the underlying proxy cryptosystem should satisfy some of the relevant properties such as collusion resistant, non-transitivity and unidirectionality that have impact on the security of the group key management scheme. We show that all schemes based on the proxy cryptography are insecure against active outsider adversary and also, we show that proxy re-encryption schemes they employ do not satisfy important desirable properties. We emphasize that the practical application of group key management schemes require their security under the active outsider adversary model.**
***Keywords*:** secure group communication, key management, proxy cryptography, active outsider adversary, forward secrecy, backward secrecy.

I. Introduction

Secure group communication is an inherent requirement for most of the collaborative applications [1]. To secure group communications the messages communicated within the group should be encrypted. The group users share a common key called as the group key. All the communications taking place within the group are encrypted using this group key. In the lifetime of a group, a new user may join the group or an existing user from the group might leave. During these group membership changing events, the current group key of the group should be changed to ensure the confidentiality of the past and the future group messages. The process of updating the group key on membership change is called rekeying. How to change the group key and communicate the changed group key to the changed group members is the central question in the key management problem. Any key management scheme should satisfy two basic security requirements viz., backward secrecy (a newly joining user should not be able to read the group messages exchanged prior to its joining) and forward secrecy (a leaving group user should not be able to read the group messages exchanged post its departure). Any key management scheme for the secure group communication should possess the qualities like minimal storage at users and group controller, low rekeying cost defined with respect to the number of encryptions and rekey messages [2].

A. Adversarial Models

The existing GKM schemes adopted the passive adversary model in which an adversary is only permitted to join and leave the group with the intent of hampering the security of the group communication. Under this model, the forward and backward secrecy security definition were provided. The existing schemes have focused on showing that the key management schemes conform to the forward and backward secrecy requirements based on passive adversary model. Another adversary model is strong adversary model formalized by Xu [3, 4], also referred to as the strong active attack model or active outsider attack model. In strong outsider attack model, an adversary is given the additional capability of compromising a legitimate user of the group. For a GKM scheme to be secure, the forward and backward secrecy requirements should be satisfied under this strong active adversary model. It is believed that by means of compromising a valid member of the group, the active adversary would obtain the current group key of the group. However, it has been shown that based on the rekeying mechanism that has been employed by the GKM schemes, an active adversary apart from the current group key, will obtain the prior group keys of the group, which may actually be no longer possessed by the compromised group user. This renders the

GKM scheme impractical. In the next section, we illustrate the active adversary with polynomial based GKM scheme as the base scheme.

B. Illustration of Strong Active Adversary Model

To illustrate the strong active adversary, we consider a GKM scheme based on access control polynomials. Suppose $U = \{u_1, u_2, \ldots, u_n\}$ be the set of users and a trusted group controller GC. Assume that each user u_i has a shared secret k_i with the GC. To securely communicate, the users needs to be given a group key. The GC chooses randomly a key K and constructs the following:

$$p(x) = (x - k_1)(x - k_2)(x - k_3) \ldots (x - k_n) + K$$

and broadcasts $p(x)$. Each user u_i evaluates $p(x)$ at k_i i.e. computes $p(k_i)$ and obtains K. The users $u_1, u_2, \ldots, u_n$ then can communicate securely among themselves using K.
Now consider a scenario wherein a new user u_{n+1} wants to join the group. GC chooses k_{n+1} and gives it securely to u_{n+1}. The group key K should be changed to ensure backward secrecy. GC chooses a new key K' and computes:

$$p'(x) = (x - k_1)(x - k_2) \ldots (x - k_n)(x - k_{n+1}) + K'$$

and broadcasts $p'(x)$. Each user $u_i \in \{1, 2, \ldots, n+1\}$ computes $p(k_i)$ and obtains K'. All the users $u_1, \ldots, u_{n+1}$ and erase K'. Each user u_i will have only keys k_i and K'.
Now consider an event of user u_2 leaving the group. The current group key K' should be changed to ensure forward secrecy. GC chooses the new key K'' and computes:

$$p''(x) = (x - k_1)(x - k_3) \ldots (x - k_n)(x - k_{n+1}) + K''$$

and broadcasts $p''(x)$. Each user $u_1, u_3, \ldots, u_{n+1}$ computes $p''(x)$ and obtains K''.
Note that polynomials $p(x)$, $p'(x)$ and $p''(x)$ are broadcasted and an adversary $\mathcal{A}$ is having access to these broadcast messages.
Now, suppose adversary $\mathcal{A}$ compromises user u_i. By compromising it, $\mathcal{A}$ gets K and K''. So, all the group messages which have been encrypted with K'' can thus be accessed by $\mathcal{A}$. Note that u_i had erased K and K', the past group keys. However, we show that $\mathcal{A}$ can obtain the keys K and K'' also. Since $\mathcal{A}$ has access to $p'(x)$, A can compute $p'(k_1)$ to obtain K' and also can compute $p(k_1)$ to obtain k. So, by compromising the legitimate user u_i of the group, $\mathcal{A}$ not only gets access to the current group but also to the past group keys though the *compromised user has erased the past group keys.*

C. The Storage and Re-keying Cost of the Scheme

Storage at user : private shared keys and group key.
Storage at GC : n keys shared with n users.
Re-key message size on join : $O(n)$ (precisely n coefficients)
Re-key message size on leave : $O(n)$ (precisely n coefficients)

D. Observations

Adversary $\mathcal{A}$ was able to obtain the past group keys because he had access to k_1 the shared secret key of u_1 with the GC. The shared secret key k_1 of user u_1 is not changed during any join and leave events. So, by having access to all the broadcast group messages and the shared secret key k_1, $\mathcal{A}$ was able to obtain the past group keys, in addition to trivially obtaining the current group key. *It should be noted that, $\mathcal{A}$ was able to acquire all the past group keys even though all of these past group keys had been erased by the user u_1.* The scheme cannot be used in practice as the scheme cannot satisfy the backward secrecy security requirement and the active adversary model is the realistic model.

II. Background

There were several attempts made to improve the performance of the secure GKM scheme. Network simulations were performed by Manz et al. [5] to compare the performance of various schhemes in a real world scenario. One of the efficient scheme which has $O(\log n)$ rekeying cost is proposed by Wong et al. [6] based on the logical key hierarchy (LKH) tree data structure. Rafaeli et al. [2] categorized the key management scheme in three broad classes: centralized, decentralized, and contributory. The LKH based scheme is a centralized GKM scheme. The details of the some of the existing centralized, decentralized and contributory GKM schemes are given by Rafaeli et al. [2]. The existing key management schemes are based on logical tree data structures, polynomial interpolation techniques, number theoretic based techniques, combinatorial design based techniques, access control polynomial based techniques, proxy cryptography, broadcast encryption techniques, gcd based methods, Elliptic Curve based techniques, etc.
Secure group communication is a need for several applications using wired networks [2], wireless networks and wireless sensor networks [7, 8, 9].
Logical tree data structures based key management schemes such as logical key tree [6], binomial tree [10], one way function tree [11] provide better performance and are scalable as the rekeying cost is $O(\log n)$. However, these schemes are shown to be insecure against the active outsider adversary [4, 3, 12]. The group key management schemes based on polynomial interpolation technique and access control polynomial based techniques require the rekey messages of $O(n)$ [13]. This requires more network bandwidth for rekeying. The schemes based on polynomial interpolation technique are shown to be insecure against the active outsider adversary model by Purushothama et al. [12]. The number theoretic based techniques such as Chinese remainder theorem [14, 13, 15, 16, 17, 18], require more computations to be carried out at group controller. The schemes have been shown to deviate from the security requirements of active outsider adversary model by Purushothama et al. [19]. The combinatorial design based technique based on exclusion basis systems by Eltoweissy et al. [20] requires lesser storage at group controller, less rekey messages for rekeying as compared to the logical tree based key management scheme. However, the scheme is shown to be insecure against the active adversary model by Purushothama et al. [12]. Broadcast encryption techniques such as in [21] requires the encryptions and rekey messages in linear with the number of group users.
Several secure GKM schemes have been proposed for wireless sensor networks [7, 8, 9, 22, 23, 24] and mobile ad-hoc

networks [25, 26, 27, 28]. Several of these schemes for wireless sensor networks are shown to be insecure against the active outsider adversary model by Purushothama et al. [29] and Chaudhari et al. [30].
So, analysis of the GKM schemes under active outsider model is very important before deploying the scheme in applications. The focus of this paper is to analyse the GKM schemes based on proxy cryptography. Recently, the proxy cryptography is used for various applications including for key management. There are several schemes proposed based on proxy cryptography [31, 32, 33, 32, 34, 35]. Proxy re-encryption(PRE) is a method of transforming a ciphertext of Alice to the ciphertext of Bob such that the proxy will not learn any information about the underlying message [36]. Also, a PRE scheme should satisfy some of the properties to be used in applications. Ateniese et al. [36] have listed the properties of a PRE scheme. The proxy cryptosystem on which the GKM schemes rely should satisfy some of the properties such as collusion resistance, non-transitivity, unidirectional etc. for the GKM scheme to be secure. In Section III, we brief about the PRE and the desirable properties that any PRE scheme should satisfy.

A. Our Contributions

The following are the contributions of this paper.

1. We review the existing key management schemes based on proxy cryptography, proposed by Hur et al. [31], Han et al. [33], Chen et al. [32], Huang et al. [34], Wang et al. [35] and Mukherjee et al. [37] for their security.
2. We focus on the PRE schemes used by the GKM schemes, and analyse for the properties that the base PRE scheme satisfy.We show that the schemes does not satisfy the crucial desirable properties of a PRE scheme.
3. We analyze the key management schemes based on proxy cryptography for their security under active outsider adversary model and show that the schemes based on proxy cryptography are not secure against active outsider adversary model.

B. Organization of the Paper

The rest of the paper is organized as follows. In Section III, we brief about the PRE scheme and the desirable properties any PRE scheme should satisfy. In each of the Sections IV, V, VI, VII, VIII and IX we elaborate on one GKM scheme using the concepts of PRE. We elaborately give the system model under which these schemes operate and comment on the properties of the PRE scheme. The security of these schemes under the active outsider adversary model is also analyzed in each of these sections. Finally we provide the conclusions in Section X.

III. Proxy Re-encryption and its Desirable Properties

In this section, we brief about PRE and the desirable properties that a PRE scheme should satisfy.

In public key cryptosystem, the sender Alice and Receive Bob will have their own public and secret key pairs. Alice will use the public key of Bob to encrypt any message to Bob and Bob will use his secret key to decrypt the ciphertext to obtain the message. Suppose Alice (delegator) wants to delegate the task of decrypting the ciphertext that were encrypted to her using her public key, to Bob (delegatee), then Alice can enable this by giving her secret key to Bob. However, in this case, Alice should keep enormous trust on Bob. Challenge is that how can Alice delegate the task of decryption of her ciphertexts to Bob without giving the secret key to Bob? This can be achieved using the PRE method. Using the PRE, the proxy is given a re-encryption key and a re-encryption procedure, wherein the ciphertext intended for Alice can be re-encrypted so that it can be decrypted by the Bob without the message being read by the proxy. Few of the applications of the PRE are email forwarding, law enforcement, carrying out the cryptographic operations on the resource constrained devices, providing access control to the outsourced data, etc [36] and key management [38, 31, 32, 33, 39, 40, 41, 34, 42]. The notion of proxy-cryptography was proposed by Blaze et al. [43]. Later, there were several proposals on PRE [44, 45, 46]. Ateniese et al. [36] formalized the notion of PRE. Also, they have listed the desirable properties of any PRE scheme. The following are the desirable properties of the PRE scheme listed by Ateniese et al. [36].

1. Unidirectional: A delegation from delegator X to delegatee Y, does not permit the delegation of decryption rights from Y to X, i.e. given $rk_{X \to Y}$ it is not possible to compute $rk_{Y \to X}$.
2. Non-interactive : Computation of the re-encryption key $rk_{X \to Y}$ can be performed by X without interaction with delegatee Y or a third party, i.e. sk_Y is not required in the computation of $rk_{X \to Y}$.
3. Proxy invisible : The existence of a proxy is concealed from the delegatee, i.e. the delegatee Y cannot differentiate the ciphertext produced by encryption under pk_Y from a ciphertext produced by re-encryption using $rk_{X \to Y}$ for some delegator X.
4. Key optimal: The storage overhead required at a delegatee to accept delegations should be constant regardless of the number of delegations accepted.
5. Original access : The delegator X can decrypt the ciphertexts originally encrypted under pk_X even after their re-encryption.
6. Collusion safe : A collusion between the proxy and a valid delegatee does not disclose the secret key of the delegator, i.e. it is impossible to compute sk_X given both $rk_{X \to Y}$ and sk_Y.
7. Non-transitive : The proxy independently cannot re-delegate the decryption rights to a third party, i.e. $rk_{X \to Z}$ computation is not possible for a proxy possessing $rk_{X \to Y}$ and $rk_{Y \to Z}$.
8. Non-transferable: The proxy, colluding with one or more valid delegatees cannot re-delegate the decryption

rights to an unauthorized third party, i.e. given $rk_{X\to Y}$, sk_Y and pk_Z, it is not possible to compute $rk_{X\to Z}$.

9. Temporary : The delegation of decryption rights is not permanent and is valid only for a specified time frame.

IV. Security Analysis of Decentralized Group Key Management Scheme for Dynamic Networks using Proxy Cryptography

The scheme of GKM proposed by Hur et al. [31] uses proxy cryptography for the rekeying messages to update the group key and securely communicate it to the valid members of the group when the group membership changes. The communication of messages within the group then takes place using the group key shared among all users within the group. The PRE scheme used for this purpose is described in the next section, followed by the GKM scheme with user join and leave scenarios in the subsequent sections.

A. PRE Scheme and its Properties

In this section, we describe the PRE scheme employed for GKM and highlight its properties. We consider a scenario wherein a user X is the delegator and user Y is the delegatee in the PRE scheme. Each user U possesses a public private key pair (pk_U, sk_U). The key generation, encryption and decryption follow the algorithms of the Elgamal scheme. The Elgamal scheme uses two primes p and q such that $p = 2q+1$ and a generator g of the group Z_p^*. The following are the algorithms used in the PRE scheme.

1. Encryption: To encrypt a message M for user X, a random number r is chosen and the ciphertext is obtained using the public key as follows:

$$(C_{X_1}, C_{X_2}) = (g^r, M.pk_X^r) = (g^r, M.g^{r.sk_X})$$

2. Decryption: To decrypt a ciphertext encrypted under the public key of X, the secret key of X is used as follows:

$$M = \frac{C_{X_2}}{(C_{X_1})^{sk_X}} = \frac{M.g^{r.sk_X}}{(g^r)^{sk_X}}$$

3. Re-encryption key: To delegate the decryption rights of a ciphertext encrypted for X to the user Y, the re-encryption key generated is

$$rk_{X\to Y} = (sk_Y - sk_X)\ mod\ p$$

4. Re-encryption: The actual re-encryption of a ciphertext encrypted for X to a ciphertext which can be decrypted by Y proceeds as follows:

$$(C_{Y_1}, C_{Y_2}) = (C_{X_1}, C_{X_2}.(C_{X_1})^{rk_{X\to Y}}) = (g^r, g^{r.sk_Y})$$

The user Y can then decrypt the ciphertext (C_{Y_1}, C_{Y_2}) using sk_Y.

1) Properties of the PRE Scheme

We analyse the aforementioned re-encryption scheme on basis of the desirable properties of a PRE scheme as proposed by Ateniese et al. [36].

1. Interactive: Computation of the re-encryption key for a delegation requires the secret key of the delegatee i.e. $rk_{X\to Y}$ requires sk_Y.

2. Bidirectional: Knowledge of $rk_{X\to Y}$ allows for the computation of $rk_{Y\to X}$ since $rk_{Y\to X} = -rk_{X\to Y}\ mod\ p$.

3. Key optimal: The delegatee Y is not required to store any additional keys to accept a delegation enabled by $rk_{X\to Y}$ and can decrypt the re-encrypted ciphertext using its personal secret key sk_Y.

4. Not collusion safe: A collusion of the proxy and the delegatee provides them to have access to $rk_{X\to Y}$ and sk_Y, thus allowing for the computation of the secret of the delegator X. The proxy and delegatee can collude to compute $(sk_Y - rk_{X\to Y})\ mod\ p = (sk_Y - (sk_Y - sk_X))\ mod\ p = sk_X\ mod\ p$.

5. Transitive: A proxy possessing $rk_{X\to Y}$ and $rk_{Y\to Z}$ can compute $rk_{X\to Z}$ without the delegation being authorized by X. The proxy can simply compute $rk_{X\to Z} = rk_{X\to Y} + rk_{Y\to Z} = (sk_Y - sk_X) + (sk_Z - sk_Y) = (sk_Z - sk_X)\ mod\ p$.

6. Transferable: A colluding proxy and the delegatee Y, can re-delegate the decryption rights to a third party Z. The proxy and Y can compute the secret key sk_X of the delegator and thus can redelegate the decryption rights to Z by computing $rk_{X\to Z}$. A collusion of the proxy and delegatee can thus compute $rk_{X\to Y} - sk_Y + sk_Z = (sk_Z - sk_X)\ mod\ p = rk_{X\to Z}$.

7. Not temporary: The re-encryption key generated is such that it allows for re-delegation of the decryption rights as long as the secret key of the delegator does not change.

8. Proxy invisible: The same decryption key and algorithm is used by a party Y to decrypt the re-encrypted ciphertext, as is used to decrypt the ciphertext directly encrypted under the public key pk_Y of Y.

9. Original access is not allowed: Once a ciphertext of user X has been re-encrypted for a delegation using the re-encryption key $rk_{X\to Y}$, it cannot be decrypted by the delegator X.

B. Setup of the GKM Scheme

In this section, we describe the notations used in the GKM scheme by Hur et al. [31] and also describe how the PRE scheme is altered to model it to securely communicate the group key.

The system model consists of a set of users $u_1, \ldots, u_m$ and n proxies $P_1, P_2, \ldots, P_n$ such that each proxy has a subset of users as its child users. Also some of the proxies have other proxies as its child proxies. Each proxy P_i holds its proxy

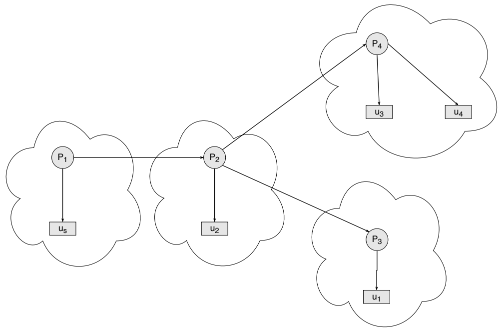

Figure. 1: System model for GKM scheme by Hur et al. [31]

key PK_i. Each PK_i is also shared with all the child users of proxy P_i as well as the child proxies of P_i. Each user u_j also possesses its public-private key pair (pk_{u_j}, sk_{u_j}). To make the system model clear, we present a specific scenario consisting of four proxy servers $P_1,\ P_2,\ P_3$ and P_4 and five users $u_1,\ u_2,\ u_3,\ u_4$ and u_s. As shown in Figure 1, each of these users is assigned to a proxy which acts as the parent proxy of the user. GK_l is the group session key at session l. When a rekey message M has to be communicated to all the group members, a user u_s acts as the sender of this message and uses the proxy key $PK_{par(u_s)}$ of its parent proxy $P_{par(u_s)}$ for encryption. The computation of this encrypted message and the communication to the group members occurs as follows:

- The sender u_s chooses a random number r_0 and computes the ciphertext.

$$(C_1^0, C_2^0) = (g^{r_0}, M.g^{(r_0+PK_{par(u_s)})GK_l})$$

- Each proxy P_i on the path from the sender to the receiver selects a random number r_i and computes the following:
$(C_1^i, C_2^i) =$
$(C_1^{par(P_i)}.g^{r_i}, C_2^{par(P_i)}.g^{(r_i-PK_{par(P_i)}+PK_i)GK_l})$
where $par(P_i)$ represents 0 when P_i is the parent proxy of the sender and represents the index of parent proxy of P_i in all other cases.

  The pair (C_1^i, C_2^i) is forwarded by the proxy P_i to each of its child proxies as well as child users.

- When a valid group member u_j receives a ciphertext from its parent proxy $P_{par(u_j)}$, it is of the following form:
$(C_1^{par(u_j)}, C_2^{par(u_j)}) =$
$(g^{r_0+\ldots+r_{par(u_j)}}, M.g^{(r_0+\ldots+r_{par(u_j)}+PK_{par(u_j)})GK_l})$
The member u_j holds the parent proxy key $PK_{par(u_j)}$ as well as the group key GK_l and thus can decrypt the ciphertext as follows to obtain the message M.

$$M = \frac{C_2^{par(u_j)}}{(C_1^{par(u_j)}.g^{PK_{par(u_j)}})^{GK_l}}$$

Consider the scenario described in Figure 1. Let the user u_4 be the receiver of the message M sent by the sender u_s. To facilitate this communication using the scheme described above, the sender computes the ciphertext.

$$(C_1^0, C_2^0) = (g^{r_0}, M.g^{(r_0+PK_1)GK_l})$$

and forwards it to the first proxy P_1 on the path from u_s to u_4. P_1 further computes a new ciphertext

$$(C_1^1, C_2^1) = (C_1^0.g^{r_1}, C_2^0.g^{(r_1-PK_0+PK_1)GK_l})$$

and forwards it to P_2 which computes

$$(C_1^2, C_2^2) = (C_1^1.g^{r_2}, C_2^1.g^{(r_2-PK_1+PK_2)GK_l})$$

The proxy P_4 receives this ciphertext (C_1^2, C_2^2) from its parent proxy P_2 and computes

$$(C_1^4, C_2^4) = (C_1^2.g^{r_4}, C_2^2.g^{(r_4-PK_2+PK_4)GK_l})$$

which is then forwarded to its child user u_4. The ciphertext received by u_4 is
$(C_1^4, C_2^4) = (g^{r_0+r_1+r_2+r_4}, M.g^{(r_0+r_1+r_2+r_4+PK_4)GK_l})$
Note that u_4 holds the proxy key PK_4 of its parent P_4 as well as the group key GK_l since u_4 is a valid member of the group and thus can compute

$$\begin{aligned}\frac{C_2^4}{(C_1^4.g^{PK_4})^{GK_l}} &= \frac{M.g^{(r_0+r_1+r_2+r_4+PK_4)GK_l}}{(g^{r_0+r_1+r_2+r_4}.g^{PK_4})^{GK_l}} \\ &= \frac{M.g^{(r_0+r_1+r_2+r_4+PK_4)GK_l}}{g^{(r_0+r_1+r_2+r_4+PK_4)GK_l}} \\ &= M\end{aligned}$$

C. Member Join

Consider a scenario wherein the current group key is GK_l. If a new group member u_{new} joins the group under parent proxy $P_{par(u_{new})}$ then the group session is updated to $l+1$ and the group key is updated to GK_{l+1} by following the steps given below.

- All legitimate members of the group knowing GK_l compute the new group key

$$GK_{l+1} = Hash(GK_l)$$

- The sender securely transmits the new group key GK_{l+1} to the new member u_{new} to facilitate group communication.
- The parent proxy $P_{par(u_{new})}$ also sends its key $PK_{par(u_{new})}$ securely to the new member u_{new} to enable u_{new} to decrypt the future group re-keying messages.

Suppose a new member u_5 joins the group described in Figure 1 under the parent proxy P_3 during the session l. The group key is updated by the existing group members u_1, u_2, u_3, u_4 and u_s by computing $GK_{l+1} = Hash(GK_l)$. The sender u_s also communicates the updated group key by computing $Enc(GK_{l+1}, pk_{u_5})$ to u_5 securely. The newly joined member u_5 also receives its parent proxy's key PK_3 by receiving $Enc(PK_3, pk_{u_5})$ from P_3 securely.

D. Member Leave

When a member u_{leave} with parent proxy $P_{par(u_{leave})}$ leaves the group after the session l, the re-keying procedure proceeds as follows:

- $P_{par(u_{leave})}$ chooses a new proxy key $PK'_{par(u_{leave})}$ and distributes it securely to all its child proxies and users excluding u_{leave}.
- The sender now generates a new group key GK_{l+1} which is independent of the previous group key and communicates GK_{l+1} using proxy cryptography to all the valid members of the group. This communication proceeds as described in Section IV-B wherein the secret message M is replaced by GK_{l+1} by the sender.

Consider the group scenario depicted in Figure 1 with the group key GK_l. Suppose the member u_4 leaves the group, the group session key is updated by the sender to GK_{l+1} which is independent of GK_l. Also, the parent proxy P_4 of the departing member updates its proxy key PK_4 to PK'_4. This updated proxy key is sent securely to its child member u_3 by communicating $Enc(PK'_4, pk_{u_3})$ to it. Sender now computes the ciphertext

$$(C_1^0, C_2^0) = (g^{r_0}, GK_{l+1}.g^{(r_0+PK_1)GK_l})$$

and forwards it to the its parent proxy P_1. P_1 then computes

$$(C_1^1, C_2^1) = (C_1^0.g^{r_1}, C_2^0.g^{(r_1-PK_0+PK_1)GK_l})$$

and forwards it to P_2 which computes

$$(C_1^2, C_2^2) = (C_1^1.g^{r_2}, C_2^1.g^{(r_2-PK_1+PK_2)GK_l})$$

This ciphertext is forwarded by P_2 to its child proxies P_3 and P_4 as well as its child user u_2. P_3 computes the ciphertext

$$(C_1^3, C_2^3) = (C_1^2.g^{r_3}, C_2^2.g^{(r_3-PK_2+PK_3)GK_l})$$

which is received by its child user u_1. Similarly, P_4 computes its ciphertext as

$$(C_1^4, C_2^4) = (C_1^2.g^{r_4}, C_2^2.g^{(r_4-PK_2+PK'_4)GK_l})$$

and forwards it to its child user u_3. Each valid user of the group, possessing the proxy key of its parent proxy as well as the group key GK_l of the prior session can thus decrypt the ciphertext received by it as shown in Section IV-B. Note that the departed user u_4 does not possess the updated proxy key PK'_4 and thus cannot obtain the updated group key GK_{l+1}.

E. Analysis of the Scheme under Strong Adversarial Model

In this section we analyse the GKM scheme proposed by Hur et al. [31] under the strong adversarial model with the help of a scenario. Consider the group configuration as shown in Figure 1. Suppose that the user joins under the u_5 joins under the proxy P_4 during session $l-1$. The session thus changes to l and the group key has to be updated to GK_l. The re-keying is performed as follows:

- The group key is updated by computing $GK_l = Hash(GK_{l-1})$. Note that the prior group key GK_{l-1} is now erased by all the group members.
- User u_5 securely receives the new group key GK_l and parent proxy key PK_4 through the following messages

$$Enc(GK_l, pk_{u_5}) \tag{1}$$

$$Enc(PK_4, pk_{u_5}) \tag{2}$$

Now suppose member u_3 leaves the group. Parent proxy P_4 will update its proxy key to PK'_4. The updated proxy key PK'_4 is also communicated to u_4 and u_5 as follows:

$$Enc(PK'_4, pk_{u_4}) \tag{3}$$

$$Enc(PK'_4, pk_{u_5}) \tag{4}$$

Further, the sender u_s updates the group session key to GK_{l+1} and communicates it to the group members u_1, u_2, u_4 and u_5 using proxy cryptography. The member u_2 receives

$$(C_1^2, C_2^2) = (g^{r_0+r_1+r_2}, GK_{l+1}.g^{(r_0+r_1+r_2+PK_2)GK_l}) \tag{5}$$

Member u_1 receives the ciphertext

$$(C_1^3, C_2^3) = (g^{r_0+r_1+r_2+r_3}, GK_{l+1}.g^{(r_0+r_1+r_2+r_3+PK_3)GK_l}) \tag{6}$$

Members u_4 and u_5 receive the ciphertext

$$(C_1^4, C_2^4) = (g^{r_0+r_1+r_2+r_4}, GK_{l+1}.g^{(r_0+r_1+r_2+r_4+PK'_4)GK_l}) \tag{7}$$

Each member u_1, u_2, u_4 and u_5 also erases the group key GK_l of the prior session.
If a new member u_6 further joins under the parent proxy P_2. The rekeying follows the following procedure

- The existing group members u_1, u_2, u_4 and u_5 update the group session key to $GK_{l+2} = Hash(GK_{l+1})$.
- The new group key is communicated to the member u_6 with the message

$$Enc(GK_{l+2}, pk_{u_6}) \tag{8}$$

Note that each of the current group member u_1, u_2, u_4, u_5 and u_6 maintains only the current group key GK_{l+2} and all the prior group keys $GK_{l-1},\ GK_l,\ GK_{l+1}$, have been erased and thus not available to them. Now suppose the member u_5 is compromised by an adversary during the session $l+2$, the adversary trivially obtains sk_{u_5} and GK_{l+2} which correspond to the compromised member's secret key and the current group session key respectively. Using the secret key of u_5, the adversary can decrypt the prior group rekeying message (1) to obtain the group key GK_l of the session l. Using the same key sk_5, the adversary can decrypt the message (4) to obtain the updated parent proxy key PK'_4. Having obtained GK_l and PK'_4, the adversary can further decrypt (7) to obtain GK_{l+1}. The adversary can obtain the group keys of the previous sessions even though they were not held by the member u_5 when it was compromised. Thus an adversary can compromise a group member and not only obtain the sensitive information currently held by the member but also based on the prior broadcast messages, the adversary can gain knowledge about the group key of prior sessions wherein the member was not compromised. This renders the scheme insecure against the active adversarial model.

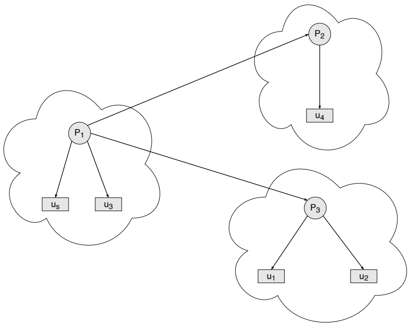

Figure. 2: System model for GKM scheme by Han et al. [33]

V. Security Analysis of Proxy Encryption based Secure Multicast Scheme in Wireless Mesh Networks

Traditionally, a PRE scheme is designed in a manner such that the proxies hold the re-encryption key needed to transform a ciphertext intended for the delegator into a ciphertext for the delegatee. However, the proxy encryption based secure multicast scheme proposed by Han et al. [33] is designed in a way such that it does not additionally require the proxies to hold any re-encryption key and rather facilitates the re-encryption by the proxies using their own secret keys. To enable this, the scheme is designed in such a manner that the delegatee is required to hold a delegation key in order to decrypt the ciphertext transformed by the proxies. The delegation key is communicated to the delegatee via the same set of proxies which will further participate in the re-encryption. The scheme designed uses the proxy encryption technique only for communicating the group keys to the group members. Further, the group communication is achieved by encrypting the group messages using the group key. In the subsequent sections, we describe the system model under consideration for this scheme followed by the working of the scheme for group communication with the scenarios of user join and leave.

A. System Model

In this section, we describe the system model for the aforementioned scheme. The scheme operates under a model consisting of a set of users $u_1, u_2, \ldots, u_m$ such that one of these users acts as the sender referred to as u_s. Each user u_j is identified by its public private key pair (pk_{u_j}, sk_{u_j}). The system also consists of a set of proxies $P_1, P_2, \ldots, P_n$. Each proxy P_i possesses its own secret key PK_i. The proxies are arranged in a topology so as to form a tree structure in such a way that some proxies act as the child proxies of others. Also each user is assigned to one proxy, which acts as its parent proxy. The system uses Elgamal based scheme for key generation, encryption and decryption. Figure 2 provides a specific group scenario consisting of users u_1, u_2, u_3, u_4 and u_s and proxies P_1, P_2 and P_3. The group topology is such that the proxy P_1 has u_s and u_3 as its child users and proxies P_2 and P_3 as its child proxies. Similarly, P_2 has u_4 as its child user. Users u_1 and u_2 are assigned as child users of P_3. We also describe how proxy encryption is used to handle the group dynamics subsequently.

B. Member Join

When a new member u_{new} joins the group under the parent proxy $P_{par(u_{new})}$ when the group session key is GK_l, the rekeying of the group is performed by following the steps given below.

- The group key is updated to $GK_{l+1} = Hash(GK_l)$ by the sender as well as each valid member of the group having access to GK_l.
- The sender securely unicasts the group key to u_{new}.
- The sender communicates the group membership delegation key to u_{new} to facilitate the future group rekeying as follows:
  1. The sender u_s computes its version of the partial key $partial\_key_{u_{new}} = (pk_{u_{new}})^{\frac{1}{sk_{u_s}}}$ and forwards it to its parent proxy.
  2. Each proxy P_i on the path from u_s to the parent proxy $P_{par(u_{new})}$ receives the $partial\_key_{u_{new}}$ from its prior proxy(or from the sender in case the proxy is the first proxy on such a path) and computes $partial\_key_{u_{new}} = (partial\_key_{u_{new}})^{\frac{1}{PK_i}}$ which is the updated partial key.
  3. The parent proxy $P_{par(u_{new})}$ computes and forwards $partial\_key_{u_{new}}$ to the newly joined user u_{new}.
  4. The $partial\_key_{u_{new}}$ received by the user u_{new} is of the form $partial\_key_{u_{new}} = (pk_{u_{new}})^{\frac{1}{sk_{u_s}\prod PK_i}}$ such that each PK_i from $\prod PK_i$ represents the proxy key of a proxy P_i on the path from the sender to u_{new}.
  5. The user u_{new} further computes the delegation key as $delegation\_key_{u_{new}} = (partial\_key_{u_{new}})^{\frac{1}{sk_{u_{new}}}}$. The $delegation\_key_{u_{new}}$ is thus of the form $g^{\frac{1}{sk_{u_s}\prod PK_i}}$.

Consider the group scenario of Figure 2. Suppose a new user u_5 joins the group under parent proxy P_2 during the session l of the group communication, then the group key has to be updated to GK_{l+1}. The existing group members u_1, u_2, u_3, u_4 and u_s update the group key by computing $GK_{l+1} = Hash(GK_l)$. Sender u_s computes $Enc(GK_{l+1}, pk_{u_5})$ and communicates it to u_5 who decrypts it to obtain GK_{l+1}. Further, sender u_s also computes $partial\_key_{u_5} = (pk_{u_5})^{\frac{1}{sk_{u_s}}}$ and forwards it to P_1. Proxy P_1 updates the partial key by computing

$$partial\_key_{u_5} = (partial\_key_{u_5})^{\frac{1}{PK_1}} = (pk_{u_5})^{\frac{1}{sk_{u_s}.PK_1}}$$

Proxy P_2 receives the $partial\_key_{u_5}$ computed by P_1 and computes the new partial key as

$$partial\_key_{u_5} = (partial\_key_{u_5})^{\frac{1}{PK_2}} = (pk_{u_5})^{\frac{1}{sk_{u_s}.PK_1.PK_2}}$$

This $partial\_key_{u_5}$ is then forwarded to the newly joined user u_5. User u_5 computes the

$$\begin{aligned} delegation\_key_{u_5} &= (partial\_key_{u_5})^{\frac{1}{sk_{u_5}}} \\ &= ((pk_{u_5})^{\frac{1}{sk_{u_s}.PK_1.PK_2}})^{\frac{1}{sk_{u_5}}} \\ &= (g^{sk_{u_5}})^{\frac{1}{sk_{u_5}.sk_{u_s}.PK_1.PK_2}} \\ &= g^{\frac{1}{sk_{u_s}.PK_1.PK_2}} \end{aligned}$$

C. Group Key Communication

The communication of rekeying messages within the group also takes place with the help of proxy cryptography. Each proxy on the path from the sender to the receiver participates in transforming the ciphertext to a form which can be decrypted by its child users. Such a group rekeying is initiated by a group membership change when a member departs from the group. To perform the rekeying, the sender selects a new group key GK_{l+1} independent of the prior group key GK_l of the group session l. The system then proceeds through the following rekeying procedure.

- The sender encrypts GK_{l+1} using its own public key with the Elgamal encryption scheme by computing

$$(C_1^0, C_2^0) = (g^{k.sk_{u_s}}, GK_{l+1}.Z^k)$$

  and forwards this ciphertext pair to its parent proxy.

- Each proxy P_i receives the ciphertext pair from its parent proxy(or from the sender in case the proxy is sender's parent proxy) and updates the ciphertext by transforming it as follows:

$$(C_1^{par(P_i)}, C_2^{par(P_i)}) = ((C_1^{par(P_i)})^{PK_i}, C_2^{par(P_i)})$$

  The updated ciphertext is forwarded to all its child proxies and group members.

- The ciphertext received by a group member u_j from its parent proxy $P_{par(u_j)}$ is of the form

$$(C_1^{par(u_j)}, C_2^{par(u_j)}) = (g^{k.sk_{u_s}\prod PK_i}, GK_{l+1}.Z^k)$$

  such that each PK_i from $\prod PK_i$ represents the proxy key of a proxy P_i on the path from the sender to u_j.

- Each group member u_j possessing its delegation key follows the subsequent sequence of steps for decryption.

  1. Let $del\_key_{u_j} = delegation\_key_{u_j}$. The group member computes

$$\begin{aligned} e(del\_key_{u_j}, C_1^{par(u_j)}) &= e(g^{\frac{1}{sk_{u_s}\prod PK_i}}, g^{k.sk_{u_s}\prod PK_i}) \\ &= e(g,g)^k \\ &= Z^k \end{aligned}$$

  2. Further, the decryption of the ciphertext to obtain the group key occurs as follows

$$GK_{l+1} = \frac{C_2^{par(u_j)}}{Z^k} = \frac{GK_{l+1}.Z^k}{Z^k}$$

Consider the group scenario from Figure 2 consisting of users u_1, u_2, u_3, u_4 and u_s. Suppose that a group key GK_{l+1} has to be communicated to the user u_4. To facilitate this communication, the sender u_s selects a random number k as per the Elgamal encryption scheme and generates the ciphertext

$$(C_1^0, C_2^0) = (g^{k.sk_{u_s}}, GK_{l+1}.Z^k)$$

and forwards it to its parent proxy P_1. The proxy P_1 computes

$$(C_1^1, C_2^1) = ((C_1^0)^{PK_1}, C_2^0) = (g^{k.sk_{u_s}.PK_1}, GK_{l+1}.Z^k)$$

This ciphertext pair (C_1^1, C_2^1) is received by P_2 and it further computes

$$\begin{aligned} (C_1^2, C_2^2) &= ((C_1^1)^{PK_2}, C_2^1) \\ &= (g^{k.sk_{u_s}.PK_1.PK_2}, GK_{l+1}.Z^k) \end{aligned}$$

and forwards this ciphertext pair to the user u_4. Note that u_4 possesses its $delegation\_key_{u_4} = g^{\frac{1}{sk_{u_s}.PK_1.PK_2}}$ and thus can compute

$$\begin{aligned} e(delegation\_key_{u_4}, C_1^2) &= e(g^{\frac{1}{sk_{u_s}.PK_1.PK_2}}, g^{k.sk_{u_s}.PK_1.PK_2}) \\ &= e(g,g)^k \\ &= Z^k \end{aligned}$$

Further, the user u_4 can perform the decryption to obtain the group key as follows

$$GK_{l+1} = \frac{C_2^2}{Z^k} = \frac{GK_{l+1}.Z^k}{Z^k}$$

D. Member Leave

When an existing group member u_{leave} with the parent proxy $P_{par(u_{leave})}$ departs from the group when the group key is GK_l, the rekeying procedure required to securely update and communicate the group key GK_{l+1} in order to maintain the forward secrecy is described below.

- The sender selects a random value ρ and encrypts it as follows

$$(C_1^0, C_2^0) = (g^{k.sk_{u_s}}, \rho.Z^k)$$

- The sender generates a revocation list which contains the identity of the user to be revoked.

- The ciphertext generated for ρ as well as the revocation list is communicated to each proxy using the same proxy encryption strategy as followed in the group key communication in Section V-C.

- Each proxy P_i other than the parent proxy of u_{leave} receives $(C_1^{par(P_i)}, C_2^{par(P_i)})$ from its parent proxy(or the sender), performs the ciphertext transformation and forwards it to its child members for decryption with the same strategy as in Section V-C. Each of these group members thus decrypts the ciphertext to obtain ρ. Let $P_x = P_{par(u_{leave})}$ represent the parent proxy of the departing user u_{leave}. Proxy P_x performs the rekeying by following the subsequent sequence of steps.

  1. Proxy P_x computes $C_1^x = (C_1^{par(P_x)})^{PK_x}$

2. For each user u_j which has P_x as its parent proxy, excluding the departing user u_{leave}, the proxy computes $C^x_{2_{u_j}} = (C_2^{par(P_x)}).g^{r.sk_{u_j}}$ for a randomly chosen r. The proxy P_x further computes and broadcasts the multiparty ciphertext

$$(C^x_1, g^r, C^x_{2_{u_j}})$$

- Each valid member u_j having P_x as its parent proxy receives the multiparty ciphertext, extracts its components $(C^x_1, g^r, C^x_{2_{u_j}})$ and uses its secret key sk_{u_j} to compute

$$\frac{C^x_{2_{u_j}}}{(g^r)^{sk_{u_j}}} = \frac{(C_2^{par(P_x)}).g^{r.sk_{u_j}}}{g^{r.sk_{u_j}}} = C_2^{par(P_x)} = C_2^0 = \rho.Z^k$$

- Let $del\_key_{u_j} = delegation\_key_{u_j}$. Further, the member evaluates

$$e(del\_key_{u_j}, C_1^x) = e(g^{\frac{1}{sk_{u_s}\prod PK_i}}, g^{k.sk_{u_s}\prod PK_i}) = e(g,g)^k = Z^k$$

and computes ρ by evaluating

$$\frac{C_2^0}{Z^k} = \frac{\rho.Z^k}{Z^k} = \rho$$

- On retrieving ρ, the group key is updated by each valid non-revoked group member by computing

$$GK_{l+1} = Hash(GK_l||\rho)$$

Note that the departed user does not receive a ciphertext which contains the encrypted value of ρ and thus cannot update the group key to GK_{l+1}.

Consider the group scenario depicted in Figure 2 consisting of 5 users u_1, u_2, u_3, u_4 and u_s. Suppose the user u_2 now leaves the group after session l then the group key has to be updated to GK_{l+1}. To facilitate this, the sender u_s selects a random value ρ and computes

$$(C_1^0, C_2^0) = (g^{k.sk_{u_s}}, \rho.Z^k)$$

and forwards it to its parent proxy P_1 along with the revocation list consisting of user u_2. The proxy P_1 further identifies that none of its child proxies are in the revoked list and thus computes

$$(C_1^1, C_2^1) = ((C_1^0)^{PK_1}, C_2^0) = (g^{k.sk_{u_s}.PK_1}, \rho.Z^k)$$

which is forwarded to its child proxies P_2 and P_3 as well as its child user u_3. User u_3 computes

$$e(deldegation\_key_{u_3}, C_1^1) = e(g^{\frac{1}{sk_{u_s}.PK_1}}, g^{k.sk_{u_s}.PK_1}) = e(g,g)^k = Z^k$$

and further obtains $\rho = \frac{\rho.Z^k}{Z^k}$.The proxy P_2 computes

$$(C_1^2, C_2^2) = ((C_1^1)^{PK_2}, C_2^1) = (g^{k.sk_{u_s}.PK_1.PK_2}, \rho.Z^k)$$

and forwards it to its child user u_4. Let $del\_key_{u_4} = delegation\_key_{u_4}$. User u_4 computes

$$e(del\_key_{u_4}, C_1^2) = e(g^{\frac{1}{sk_{u_s}.PK_1.PK_2}}, g^{k.sk_{u_s}.PK_1.PK_2}) = e(g,g)^k = Z^k$$

and subsequently computes $\rho = \frac{\rho.Z^k}{Z^k}$. The proxy P_3 identifies the revoked user as one of its child users and computes

$$C_1^3 = (C_1^1)^{PK_3} = g^{k.sk_{u_s}.PK_1.PK_3}$$

P_3 also selects a random number r, computes

$$C^3_{2_{u_1}} = (C_2^1).g^{r.sk_{u_1}} = \rho.Z^k.g^{r.sk_{u_1}}$$

and broadcasts the multiparty ciphertext $(C_1^3, g^r, C^3_{2_{u_1}})$. The user u_1 receives this ciphertext and computes

$$\frac{C^3_{2_{u_1}}}{(g^r)^{sk_{u_1}}} = \frac{\rho.Z^k.g^{r.sk_{u_1}}}{g^{r.sk_{u_1}}} = \rho.Z^k$$

Let $del\_key_{u_1} = delegation\_key_{u_1}$. Further user u_1 can compute

$$e(del\_key_{u_1}, C_1^3) = e(g^{\frac{1}{sk_{u_s}.PK_1.PK_3}}, g^{k.sk_{u_s}.PK_1.PK_3}) = e(g,g)^k = Z^k$$

Finally, u_1 computes $\rho = \frac{\rho.Z^k}{Z^k}$. The group key for session $l+1$ is hence computed as $GK_{l+1} = Hash(GK_l||\rho)$. Note that the departed user u_2 is excluded by its parent proxy P_3 while generating the multiparty ciphertext and thus inhibiting it from accessing the value ρ. This preserves the forward secrecy of the group by preventing the departing user u_2 from computing the updated group key GK_{l+1}.

E. Analysis of the Scheme under the Strong Active Adversarial Model

Consider the group scenario as shown in Figure 2. If a user u_5 joins under the proxy P_2 when the group key is GK_l. Since the group experienced a membership change, the group session is updated to $l+1$ and the group key is updated to $GK_{l+1} = Hash(GK_l)$ by each valid member which was a part of the group during the session l. The updated group key is also communicated to the new joining member u_5 securely as follows

$$Enc(GK_{l+1}, pk_{u_5}) \tag{9}$$

Also the delegation key $g^{\frac{1}{sk_{u_s}.PK_1.PK_2}}$ for the user u_5 is communicated to it via proxy encryption. Note that all the group members u_1, u_2, u_3, u_4 and u_s who held the prior group key GK_l, erase this key and only retain GK_{l+1} for future group communication, along with the newly joined member u_5. Further if the user u_2 departs from the group, the sender u_s selects a new random ρ and obtains the ciphertext $(C_1, C_2) = (g^{k.sk_{u_s}}, \rho.Z^k)$. This ciphertext pair is forwarded to the proxies P_2 and P_3. Also u_s broadcasts the revocation list containing the identity of u_2. Each proxy other than the parent proxy P_3, performs the ciphertext transformation using proxy encryption and forwards it to its child users. The proxy P_1 computes the ciphertext

$$(g^{k.sk_{u_s}.PK_1}, \rho.Z^k) \tag{10}$$

and forwards it to its user u_3 and the child proxies P_2 and P_3. The users u_4 and u_5 receive

$$(g^{k.sk_{u_s}.PK_1.PK_2}, \rho.Z^k) \tag{11}$$

from their parent proxy P_2. The proxy P_3 identifies that the revocation list has its child user u_2. P_3 then computes $C_2' = (g^r, C_2.g^{r.sk_{u_1}})$. The ciphertext (C_1, C_2') is communicated to its child user u_1. User u_1 further can decrypt the ciphertext to obtain ρ. Thus each member other than u_2 receives ρ and can compute the new group key as $GK_{l+2} = Hash(GK_{l+1}||\rho)$. Note that u_2 did not receive ρ from its parent proxy P_3 and thus cannot compute the group key for session $l+2$. The group members u_1, u_3, u_4, u_5 and u_s erase the prior group key GK_{l+1} and only maintain the updated key GK_{l+2}. If user u_6 now joins the group under the parent proxy P_3. All the existing group members update their key to $GK_{l+3} = Hash(GK_{l+2})$. The updated group key GK_{l+3} is delivered to the new member u_6 securely by the sender using $Enc(GK_{l+3}, pk_{u_6})$. The group members u_1, u_3, u_4, u_5 and u_s erase the key GK_{l+2} and communicate using the updated group key GK_{l+3} which is shared with the newly joined member u_6. User u_6 also receives its group delegation key $g^{\frac{1}{sk_{u_s}.PK_1.PK_3}}$ via the proxy encryption performed by the proxies P_1 and P_3 on the path from the sender to u_6. Now consider the user u_5 is compromised during the session $l+3$. In this case the adversary trivially gets an access to the current group session key GK_{l+3}. Additionally, the adversary also acquires the secret key sk_{u_5} of the compromised user and the group delegation key $g^{\frac{1}{sk_{u_s}.PK_1.PK_2}}$. The adversary can thus decrypt the message (9) encrypted using the public key of the compromised member by using the secret key sk_{u_5} acquired by it in order to get an access to the group key GK_{l+1}. Further, the adversary can also decrypt the message (11) communicated by P_2 to u_5 to obtain the value ρ, thus allowing the computation of the $GK_{l+2} = Hash(GK_{l+1}||\rho)$. Thus, it can be observed that even though the prior group session keys were erased by a group member, the adversary can obtain them merely by using the compromised node's secret keys and recording the group rekeying messages broadcast by the sender and proxies.

VI. Security Analysis of Secure Group Key Management Scheme using Unidirectional Proxy Re-encryption Schemes

The scheme of GKM using PRE proposed by Chen et al. [32] finds an application of the RSA based PRE scheme for the purpose of key management. The scheme uses the PRE concepts for secure delivery of the group key to all the group members. The actual group communication further is achieved with encryption using the group key. The dynamic nature of the group is accounted for by the user joins and leaves permitted by the group. To maintain the basic security requirements of forward secrecy and backward secrecy, the group is required to be rekeyed on every membership change event. To facilitate the rekeying, the scheme described in this section makes use of PRE concepts. In the subsequent sections we describe the PRE scheme employed in the system, the system setup and the operation of the system. Further we also describe the group rekeying operations with respect to member join and leave events, followed by the analysis of the scheme under the strong adversarial model.

A. PRE Scheme and its Properties

In this section we describe the PRE scheme employed in the system. The scheme used is based on the RSA algorithm and uses its public and private key generation algorithms. The system parameters chosen by the group manager are as follows: Two large primes p and q are chosen and $n = pq$ is computed. The parameter n is a public parameter of the system. The group manager also computes its secret parameter $\phi(n) = (p-1)(q-1)$. Each entity U is assigned the public private keys (pk_U, sk_U). The algorithms used by the PRE scheme are described below.

1. Encryption: The encryption of a message M for user X is done as follows:

$$C_X = M^{pk_X} mod\ n$$

2. Decryption: The decryption of a ciphertext C_X encrypted under the public key of X is performed by computing

$$M = C_X^{sk_X} mod\ n$$

3. Re-encryption Key: To enable the transformation of ciphertext encrypted under the public key of X to a ciphertext decryptable with the secret key of Y, the re-encryption key $rk_{X\to Y}$ generated is

$$rk_{X\to Y} = \frac{pk_Y}{pk_X}\ mod\ \phi(n) = pk_Y.sk_X\ mod\ \phi(n)$$

4. Re-encryption: To re-encrypt the ciphertext intended for X to that for Y, the re-encryption is performed as follows

$$C_Y = C_X^{rk_{X\to Y}}\ mod\ n = M^{pk_Y}\ mod\ n$$

1) Properties of the PRE Scheme

In this section, we analyze the aforementioned PRE scheme with respect to the desirable properties of a PRE scheme put forward by Ateniese et al. [36].

1. Non-interactive: The computation of the re-encryption key $rk_{X\to Y}$ does not require the secret key sk_Y of the delegatee. Thus the re-encryption key can be obtained by the delegator without any form of an interaction with the delegatee.

2. Unidirectional: The re-encryption key $rk_{X\to Y}$ does not facilitate the computation of the $rk_{Y\to X}$ since $rk_{X\to Y}$ does not reveal any information about the secret key sk_Y of the party Y, which is required for the computation of $rk_{Y\to X}$.

3. Key optimal: The ciphertext delegated from X to Y by using the re-encryption key $rk_{X\to Y}$ allows for decryption by the party Y by using its own secret key sk_Y and does not require the delegatee to store any additional keys to accept the delegation.

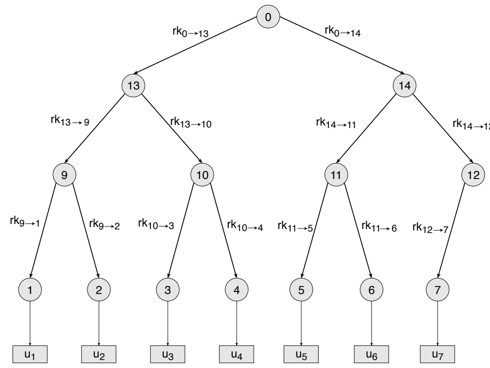

Figure. 3: System model for GKM scheme by Chen et al. [32]

4. Not collusion safe: A collusion of the proxy possessing $rk_{X\to Y}$ and the delegatee Y holding sk_Y can evaluate $sk_Y.rk_{X\to Y}\ mod\ \phi(n) = sk_Y.(\frac{pk_Y}{pk_X})\ mod\ \phi(n) = \frac{1}{pk_X}\ mod\ \phi(n) = sk_X$ and thus obtain the secret key sk_X of the delegator.

5. Transitive: A proxy holding the re-encryption key $rk_{X\to Y}$ for a delegation from X to Y and $rk_{Y\to Z}$ for a delegation from Y to Z can compute the delegation key $rk_{X\to Z} = rk_{X\to Y}.rk_{Y\to Z} = (pk_Y.sk_X).(pk_Z.sk_Y) = (pk_Z.sk_X).(pk_Y.sk_Y) = pk_Z.sk_X\ mod\ \phi(n)$ for a delegation from X to Z without the authorization of the party X.

6. Transferable: A collusion of the delegatee Y holding sk_Y and the proxy possessing the re-encryption key $rk_{X\to Y}$ can compute the secret key sk_X of the delegator as described by the collusion susceptible property. This further can be extended to the colluding proxy and delegatee being able to transfer the decryption rights to a third party Z by computing the re-encryption key $rk_{X\to Z} = pk_Z.sk_X\ mod\ \phi(n)$.

7. Not temporary: The re-encryption key enabling the delegation of decryption rights is such that it allows for re-encryption of ciphertexts from the delegator to the delegatee as long as the secret key of the delegator remains the same as that used for computing the re-encryption key.

8. Proxy invisible: The decryption key and algorithm employed in the decryption of a ciphertext directly encrypted under the public key sk_Y of party Y and the ciphertext re-encrypted from a party X to Y using the re-encryption key $rk_{X\to Y}$, are the same.

9. Original access is not allowed: A ciphertext which has been re-encrypted from a party X to Y using the re-encryption key $rk_{X\to Y}$ cannot be decrypted by the delegator X after the re-encryption.

B. Setup and Group Communication

The system consists of a group manager who manages the group topology and the keying and rekeying operations of the group. The group manager creates a hierarchy of nodes in such a way that the leaf nodes are assigned to one member each of the group managed by the group manager. Each node i in the hierarchy has a public private key pair (pk_i, sk_i) assigned to it. These key pairs are also generated using the RSA based key generation algorithm. Each edge from a node i at one level in the hierarchy to a node j at the next lower level is assigned with the re-encryption key which transforms the ciphertext for the node i into ciphertext for node j. The Figure 3 shows a scenario with 7 group members and also shows the re-encryption key assigned to each edge. Each user holds the re-encryption keys on the path from the root to itself. It also holds the secret key associated with its leaf node. For example, u_1 possesses the re-encryption keys $rk_{0\to13}, rk_{13\to9}$ and $rk_{9\to1}$ and the secret key sk_1 of its leaf node. To communicate the group key of session l to the members of the group, the group manager encrypts the group key GK_l using the public key of the root node of the hierarchy as follows:

$$C_0 = Enc(GK_l, pk_0)$$

Each user performs re-encryptions consecutively with each path re-encryption key it holds on the path from the root to its leaf node and finally decrypts the ciphertext with the secret key of its leaf node in order to obtain the group key. For example, consider the scenario from the perspective of u_1. The member u_1 performs the following sequence of re-encryptions followed by decryption to obtain the group key GK_l.

$$\begin{aligned} C_{13} &= Re - Enc(C_0, rk_{0\to13}) \\ C_9 &= Re - Enc(C_{13}, rk_{13\to9}) \\ C_1 &= Re - Enc(C_9, rk_{9\to1}) \\ GK_l &= Dec(C_1, sk_1) \end{aligned}$$

C. Member Join

When a new member u_{new} joins the group after the session l, the member is first assigned to an unoccupied leaf node in the hierarchy. Further, the rekeying of the group occurs as follows.

1. Group key is updated by all the members as $GK_{l+1} = Hash(GK_l)$.

2. The key pairs (pk_i, sk_i) of each node i on the path from the root to new member are updated.

3. The re-encryption keys on this path are also updated by the group manager.

4. The secret key of the leaf node to which u_{new} is assigned is securely given to the new member.

5. GK_{l+1} is securely given to the new member using its secret key.

D. Member Leave

When a member u_{leave} departs from the group after the session l, the rekeying procedure followed by the group is described below.

1. The group manager updates the public private key pair (pk_i, sk_i) of each node i on the path from root to the leaving member.
2. The affected re-encryption keys on the path from root to the leaving member are also updated by the group manager.
3. A new group key GK_{l+1} is selected by the group manager independent of the prior group key GK.
4. GK_{l+1} is then encrypted with the new public key of the root node and broadcasted.
5. Each valid member performs re-encryptions as described in Section VI-B to obtain the group key.

E. Analysis of the Scheme under the Strong Adversary Model

The analysis of the scheme by Chen et al. [32] has been performed under the strong adversary model by Purushothama et al. [12]. For completeness purposes, we provide the analysis in this section. Consider the group scenario given in Figure 3 consisting of 7 member $u_1, \ldots, u_7$. Assume that the group key is GK_l during the session l of the group communication. Now if a new member u_8 joins the group, the group session key has to be updated to GK_{l+1}. The rekeying of the group is initiated by the group manager by first assigning the member u_8 to the node 8 in the hierarchy and associating a public private key pair (pk_8, sk_8) with this node. The secret key sk_8 is communicated securely to the new member u_8. The group key is updated by each member which was a part of the group during session l by computing $GK_{l+1} = Hash(GK_l)$. The updated group key is communicated to the new member securely with the message

$$C_8 = Enc(GK_{l+1}, pk_8) \tag{12}$$

The group manager also updates the public-private key pairs of all nodes on the path from root to the leaf node of new member u_8, that is, the key pairs of nodes 0, 14 and 12 are now updated to $(pk'_0, sk'_0), (pk'_{14}, sk'_{14})$ and (pk'_{12}, sk'_{12}). The necessary re-encryption keys are also updated as $rk_{0' \to 13}, rk_{0' \to 14'}, rk_{14' \to 11}, rk_{14' \to 12'}$ and $rk_{12' \to 7}$. Also a new re-encryption key $rk_{12' \to 8}$ is computed. Note that the re-encryption keys are public. If the member u_4 now leaves the group, the group manager updates the key pairs of nodes 0, 13 and 10 to $(pk''_0, sk''_0), (pk'_{13}, sk'_{13})$ and (pk'_{10}, sk'_{10}). The updated re-encryption keys are $rk_{0'' \to 14'}, rk_{0'' \to 13'}, rk_{13' \to 9}, rk_{13' \to 10'}$ and $rk_{10' \to 3}$. The group manager further selects a new group key GK_{l+2} independent of the prior group key and broadcasts the encrypted message

$$C''_0 = Enc(GK_{l+2}, pk''_0) \tag{13}$$

Each group member performs the transformation of ciphertext using the root to leaf path re-encryption keys followed by decryption to obtain the group key. Further if the group member u_5 leaves the group when the group key is GK_{l+3}, the group manager updates the key pairs of nodes 0, 14 and 11 to $(pk'''_0, sk'''_0), (pk''_{14}, sk''_{14})$ and (pk'_{11}, sk'_{11}). The re-encryption keys affected by these key pair updates are updated as $rk_{0''' \to 13'}, rk_{0''' \to 14''}, rk_{14'' \to 12'}, rk_{14'' \to 11'}$ and $rk_{11' \to 6}$. A new group key GK_{l+3} independent of the prior keys is selected by the group manager and

$$C'''_0 = Enc(GK_{l+3}, pk'''_0) \tag{14}$$

is broadcast in the group. Each group member can perform the re-encryptions and decryption to obtain the group key GK_{l+3}. Now consider the member u_8 is compromised by an adversary. The adversary acquires an access to sk_8 as well as the current group key GK_{l+3}. The adversary can now decrypt the message (12) to gain the group key GK_{l+1} during the session $l + 1$. Note that all the re-encryption keys published were public and accessible to the adversary. The adversary also has an access to the ciphertext C''_0 from the broadcast message (13). Using the re-encryption keys $rk_{0'' \to 14'}, rk_{14' \to 12'}$ and $rk_{12' \to 8}$, the adversary can perform the following re-encryptions.

$$C'_{14} = Re - Enc(C''_0, rk_{0'' \to 14'}) \tag{15}$$
$$C'_{12} = Re - Enc(C'_{14}, rk_{14' \to 12'}) \tag{16}$$
$$C_8 = Re - Enc(C'_{12}, rk_{12' \to 8}) \tag{17}$$

The ciphertext C_8 can now be decrypted by the adversary using sk_8 to obtain the group key GK_{l+2}. Note that the prior group keys were erased by each member after a group key update. By compromising a node, the adversary not just gains access to the current group key but also obtains the group keys used in the prior sessions.

VII. Security Analysis of Secure Multicast Scheme in Dynamic Environments

Huang et al. [34] proposed a scheme for data multicasting in an environment where the multicast group has a dynamic topology and allows for new group members to join the multicast group and the existing members to depart from the group. The crucial properties of forward and backward secrecy in a dynamic topology of a group require the group key to be changed after every member join or leave event. This scheme makes use of proxy cryptography to handle the user join and leave events in a dynamic multicast group. The GKM schemes analysed by far in this paper have the concept of a group key using which the messages intended for the group are encrypted in a way such that each valid group member possessing the group key can decrypt them to obtain the group messages. Also proxy cryptography has been used for communicating the group rekeying messages. However, the scheme proposed by Huang et al. [34] uses proxy cryptography for communicating the group messages and eliminates the concept of a shared group key among the group members. In the subsequent sections, we describe the system model and illustrate how the scheme maintains forwards and backward secrecy in the event of group membership change.

A. System Model

The scheme proposed by Huang et al. [34] operates under a model which consists of a set of proxies $P_1, P_2, \ldots, P_n$ and a multicast group comprising of a sender u_s and a set of users $u_1, u_2, \ldots, u_m$ participating in the multicast group. Each user u_j is identified by its unique public-private key pair (pk_{u_j}, sk_{u_j}). Each proxy P_i is assigned a unique private key

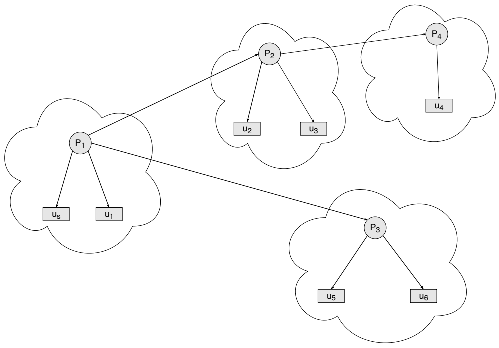

Figure. 4: System model for GKM scheme by Huang et al. [34]

PK_i. The proxies are arranged in a topology such that each proxy has a subset of the users assigned as its child users. Some proxies also have other proxies assigned as their child proxies. The network topology consisting of users and proxies resembles a tree network wherein each user has exactly one proxy assigned as its parent proxy. Also each proxy other than the parent proxy of the sender has exactly one proxy assigned as its parent. Consider a specific group scenario represented in Figure 4 consisting of users $u_1, u_2, u_3, u_4, u_5, u_6$ and u_s and the proxies P_1, P_2, P_3 and P_4. Proxy P_1 is the parent proxy of sender u_s, user u_1 and the proxies P_2 and P_3. Proxy P_2 is the parent proxy of users u_2 and u_3 and the proxy P_4. Similarly, proxy P_3 is assigned the users u_5 and u_6 as its child users and the proxy P_4 acts as the parent proxy of user u_4. In the following sections, we describe the multicast communication scheme with respect to the user join and leave events and analyze how the scheme performs under the active adversarial model.

B. Member Join

The scheme proposed is applicable to a multicast group which consists of a member u_s acting as a sender of the group and hence any user intending to join the group is required to send a join request to the sender u_s. Suppose a new user u_{new} sends a joining request with $P_{par(u_{new})}$ as its parent proxy to u_s, then u_s accepts it by sending a session identification key K_{SID} to the u_{new} securely by encrypting it with the public key $pk_{u_{new}}$ of u_{new}. The sender thus computes $Enc(K_{SID}, pk_{u_{new}})$. Further the sender is also required to communicate a key referred to as the completely composed key(CCK) to the newly joined user u_{new} in order to allow u_{new} to access the group messages. The sender u_s performs the following sequence of steps to communicate the $CCK_{u_{new}}$ to the user u_{new}.

- The sender u_s computes $PCK_{u_{new}} = sk_{u_s} + K_{SID}$ and forwards it to its parent proxy.
- Each proxy P_i on the path from the sender u_s to u_{new} receives the $PCK_{u_{new}}$ from its parent proxy(or from the sender if P_i is the parent proxy of u_s) and updates the $PCK_{u_{new}}$ by computing $PCK_{u_{new}} = PCK_{u_{new}} + PK_i$. This $PCK_{u_{new}}$ is then forwarded to its child proxies.
- The newly joined user u_{new} receives the $PCK_{u_{new}}$ from its parent proxy and computes $CCK_{u_{new}} = PCK_{u_{new}} - K_{SID} = sk_{u_s} + \sum PK_i$, for each P_i on the path from sender u_s to u_{new}.

Consider the group scenario described in Figure 4 consisting of users $u_1, u_2, u_3, u_4, u_5, u_6$ and u_s and proxies P_1, P_2, P_3 and P_4. If a new user u_7 sends a join request to u_s with proxy P_4 as the parent proxy, the following sequence of steps will be performed to accept the join request.

- Sender u_s securely communicates a session ID key K_{SID} to u_7 by computing $Enc(K_{SID}, pk_{u_7})$.
- Further, the sender u_s computes its version of the partially composed key by computing $PCK_{u_7} = sk_{u_s} + K_{SID}$ and forwards it to its parent proxy P_1.
- Proxy P_1 updates the PCK_{u_7} by computing $PCK_{u_7} = PCK_{u_7} + PK_1 = sk_{u_s} + K_{SID} + PK_1$ and forwards this key to its child proxy P_2 which is the next proxy on the path from the root to the newly joined user u_7.
- Proxy P_2 computes $PCK_{u_7} = PCK_{u_7} + PK_2 = sk_{u_s} + K_{SID} + PK_1 + PK_2$ and forwards it to P_4 which further computes $PCK_{u_7} = PCK_{u_7} + PK_4 = sk_{u_s} + K_{SID} + PK_1 + PK_2 + PK_4$ and forwards it to the user u_7.
- User u_7, on receiving PCK_{u_7}, computes $CCK_{u_7} = PCK_{u_7} - K_{SID} = sk_{u_s} + P_1 + P_2 + P_4$ which will be used to decrypt the group messages.

C. Member Leave

The scheme proposed by Huang et al. [34] does not have the concept of a commonly shared group key and rather uses proxy cryptography to allow valid group members to decrypt the group messages. Traditionally, a GKM scheme handles the departure of a group member by following a mechanism which updates the shared group key. The scheme by Huang et al. [34] however, uses a novel approach to handle the departure of a group member. If a member u_{leave} intends to depart from the group, the departure has to be conveyed to the parent proxy of the leaving member. Let $P_i = P_{par(u_{leave})}$ be the parent proxy of the leaving member u_{leave}. The departure of a member is handled locally by the parent proxy and all the other members having P_i as their parent. The sequence of steps followed for maintaining forward secrecy within the multicast communication is as follows:

- The parent proxy P_i updates its proxy key by choosing a new key PK_i'.
- Proxy P_i further computes $\delta k = PK_i' - PK_i$.
- This δk value is securely sent to each of the child users as well as child proxies of P_i by encrypting it with the private keys of each member independently.
- Each valid member u_j other than the departing member receives the updated δk and updates its CCK_{u_j} by computing $CCK_{u_j} = CCK_{u_j} + \delta k$

Consider the group scenario depicted in Figure 4. If the member u_2 departs from the group, the parent proxy P_2 updates its proxy key to PK_2' and the subsequent sequence of steps is followed to maintain forward secrecy of the group.

- Proxy P_2 computes $\delta k = PK_2' - PK_2$ and performs the encryption $Enc(\delta k, pk_{u_3})$ and $Enc(\delta k, PK_4)$ in order to communicate its updated proxy key to its child user u_3 and its child proxy P_4.
- User u_3 further updates its CCK by computing $CCK_{u_3} = CCK_{u_3} + \delta k = sk_{u_s} + PK_1 + PK_2'$.
- The proxy P_4 obtains δk and computes he ciphertext $Enc(\delta k, pk_{u_4})$ and sends it to its child user u_4.
- User u_4 now obtains δk and compute $CCK_{u_4} = CCK_{u_4} + \delta k = sk_{u_s} + PK_1 + PK_2' + PK_4$.

D. Group Communication

As mentioned in the prior sections, this scheme of multicast group communication does not use a shared group key and the group communication occurs with the help of proxy cryptography. The group communication is initiated by the sender by simply encrypting the group message M using its own public key pk_{u_s}. The sequence of steps through which the system proceeds are as follows:

- The sender selects a random number r and computes

  $$(C_1^0, C_2^0) = Enc(M, pk_{u_s}) = (g^r, M.g^{r.sk_{u_s}})$$

  and forwards it to its parent proxy.
- Each proxy P_i receives the ciphertext $(C_1^{par(P_i)}, C_2^{par(P_i)})$, where $par(P_i)$ represents the index of parent proxy of P_i(and 0 in case P_i is the parent proxy of the sender).
- Proxy P_i further computes

  $$\begin{aligned}(C_1^i, C_2^i) &= (C_1^{par(P_i)}, C_2^{par(P_i)}.(C_1^{par(P_i)})^{PK_i}) \\ &= (g^r, M.g^{r(sk_{u_s} + \sum PK_j)})\end{aligned}$$

  for each P_j on the path from the sender to the proxy P_i(including P_i). This ciphertext obtained is then forwarded by P_i to all its child proxies as well as child users.
- Each user u_j receives the ciphertext which is thus of the form $(C_1^{par(u_j)}, C_2^{par(u_j)}) = (g^r, M.g^{r.CCK_{u_j}})$ and can decrypt it by computing

  $$\frac{C_2^{par(u_j)}}{(C_1^{par(u_j)})^{CCK_{u_j}}} = \frac{M.g^{r.CCK_{u_j}}}{(g^r)^{CCK_{u_j}}} = M$$

Consider the same group scenario as depicted in Figure 4 consisting of 7 users $u_1, \ldots, u_6$ and u_s and the proxies $P_1, \ldots, P_4$. Each one of the users u_j is a part of the group since its join request has been accepted by the sender u_s and hence the user possesses its completely composed key CCK_{u_j}. User u_1 holds $CCK_{u_1} = sk_{u_s} + P_1$, users u_2 and u_3 possess CCK_{u_2} and CCK_{u_3} respectively such that $CCK_{u_2} = CCK_{u_3} = sk_{u_s} + P_1 + P_2$. Similarly, user u_4 holds the key $CCK_{u_4} = sk_{u_s} + P_1 + P_2 + P_4$. Also, users u_5 and u_6 hold the keys CCK_{u_5} and CCK_{u_6} respectively where $CCK_{u_5} = CCK_{u_6} = sk_{u_s} + P_1 + P_3$. When the sender u_s intends to communicate a message to the group members, it computes the ciphertext

$$(C_1^0, C_2^0) = (g^r, M.g^{r.sk_{u_s}})$$

This ciphertext is then communicated to the parent proxy P_1. Proxy P_1 computes

$$(C_1^1, C_2^1) = (C_1^0, C_2^0.(C_1^0)^{PK_1}) = (g^r, M.g^{r(sk_{u_s} + PK_1)})$$

and forwards this ciphertext pair to u_1 as well as the proxies P_2 and P_3. User u_1 computes

$$\frac{C_2^1}{(C_1^1)^{CCK_{u_1}}} = \frac{M.g^{r(sk_{u_s}+PK_1)}}{(g^r)^{CCK_{u_1}}} = \frac{M.g^{r(sk_{u_s}+PK_1)}}{(g^r)^{sk_{u_s}+PK_1}} = M$$

Also proxy P_2 computes

$$\begin{aligned}(C_1^2, C_2^2) &= (C_1^1, C_2^1.(C_1^1)^{PK_2}) \\ &= (g^r, M.g^{r(sk_{u_s} + PK_1 + PK_2)})\end{aligned}$$

and forwards this ciphertext pair to its child users u_2 and u_3 and the child proxy P_4. The users u_2 and u_3 can decrypt the ciphertext pair with their respective completely composed key $sk_{u_s} + PK_1 + PK_2$. Similarly, the proxy P_3 computes its ciphertext pair as

$$\begin{aligned}(C_1^3, C_2^3) &= (C_1^1, C_2^1.(C_1^1)^{PK_3}) \\ &= (g^r, M.g^{r(sk_{u_s} + PK_1 + PK_3)})\end{aligned}$$

and forwards it to its child users u_5 and u_6 who can decrypt it using the $CCK_{u_5} = CCK_{u_6} = sk_{u_s} + PK_1 + PK_3$. Also, the proxy P_4 also computes

$$\begin{aligned}(C_1^4, C_2^4) &= (C_1^2, C_2^2.(C_1^2)^{PK_4}) \\ &= (g^r, M.g^{r(sk_{u_s} + PK_1 + PK_2 + PK_4)})\end{aligned}$$

This ciphertext is received by the user u_4 who possesses $CCK_{u_4} = sk_{u_s} + PK_1 + PK_2 + PK_4$ and thus can obtain M by decrypting the received ciphertext pair. Note that each valid member of the group possesses its corresponding CCK and thus can decrypt the ciphertext corresponding to the group message M sent by the sender u_s.

E. Analysis of the Scheme under Strong Active Adversary Model

We note that, the scheme described in this section achieves multicast communication only by depending on the concepts of proxy re-cryptography. The group communication occurs without the concept of a shared group key among the members of the group. We know that the strong active adversary model focuses on the compromise of a group member to gain access to the current as well as the prior group keys. The absence of a group key in the aforementioned scheme, renders it unsuitable to be analyzed under the strong active adversary model.

VIII. Security Analysis of Group Key Management Scheme based on Proxy Re-cryptography for Near Space Networks

In this section, we describe a group communication scheme which employs the same PRE scheme as described in Sec-

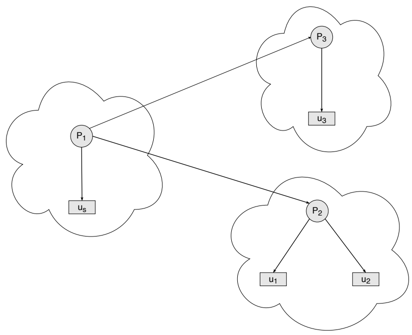

Figure. 5: System model for GKM scheme by Wang et al. [35]

tion IV-A. The PRE scheme is introduced in order to facilitate the rekeying of the group while maintaining forward and backward secrecy of the group. Only the communication of the group key to each of the valid group members occurs using the PRE concepts. The actual communication of the group messages, then takes place between users using a shared group key. This scheme of group communication makes use of proxy cryptography for communicating the group rekeying messages and also adopts the Elgamal scheme for encryption and decryption. In the subsequent section, we describe the system model under which the scheme operates.

A. System Model

The scheme proposed by Wang et al. [35] operates in a system model which consists of a set of m users $u_1, \ldots, u_m$ and n proxy servers $P_1, \ldots, P_n$. The set of users also consists of one user u_s, which acts as the sender of the messages within the group. Each user is assigned as a child user to exactly one parent proxy. Also each proxy other than the parent proxy of the sender is assigned as a child proxy to one of the proxies. Each proxy P_i possesses its proxy key PK_i. In addition to its own proxy key, each proxy P_i(excluding parent proxy of sender u_s) also holds the proxy key $PK_{par(P_i)}$ of its parent proxy $P_{par(P_i)}$. The sender also possesses a key PK_0 which is shared with its parent proxy. Also, each user u_j possesses the proxy key of its parent proxy $P_{par(u_j)}$. To make the system model clear, consider the group scenario depicted in Figure 5 consisting of a sender u_s, group members u_1, u_2 and u_3 and the proxies P_1, P_2 and P_3. The sender has proxy P_1 as its parent proxy, users u_1 and u_2 have P_2 as its parent proxy and similarly user u_3 has proxy P_3 as its parent proxy. Proxies P_2 and P_3 are assigned as child proxies to proxy P_1. The sender u_s holds the key PK_0 which is also shared with its parent proxy P_1. Proxy P_1 also holds its own proxy PK_1. Similarly, proxy P_2 holds the keys PK_1 and PK_2 and proxy P_3 holds the keys PK_1 and PK_3. Let GK_l be the group key during the session l of the group communication. The scheme is designed in such a way that the key g^{GK_l} is a public key. In the following section, we describe how the group rekeying messages are communicated by the sender to each member of the group and analyze the weakness of the scheme.

B. Communication of the Rekeying Messages

We now describe the scheme on basis of how the sender communicates the group rekeying messages to all the valid members of the group. Let M be the group rekeying message required to be communicated to each group member. The sender selects a random number r_0 and computes the ciphertext pair

$$(C_1^0, C_2^0) = (g^{r_0}, M.g^{GK_l+r_0+PK_0})$$

and communicates this to its parent proxy. Further, each proxy P_i on the path from u_s to any valid group member receives the ciphertext pair $(C_1^{par(P_i)}, C_2^{par(P_i)})$ where $par(P_i)$ returns the index of the parent proxy of proxy of P_i(0 in case P_i is sender's parent proxy). On receiving the ciphertext pair, the proxy P_i randomly selects r_i and computes

$$\begin{aligned}(C_1^i, C_2^i) &= (C_1^{par(P_i)}.g^{r_i}, C_2^{par(P_i)}.g^{r_i+PK_i-PK_{par(P_i)}}) \\ &= (g^{r_0+\ldots+r_i}, M.g^{GK_l+r_0+\ldots+r_i+PK_i})\end{aligned}$$

which is then forwarded to its child proxies as well as child users. Each user u_j receiving the ciphertext pair from its parent proxy $P_i = P_{par(u_j)}$ can further decrypt the ciphertext by computing

$$\frac{C_2^i}{C_1^i.g^{GK_l}.g^{PK_i}} = \frac{M.g^{GK_l+r_0+\ldots+r_i+PK_i}}{g^{GK_l+r_0+\ldots+r_i+PK_i}} = M$$

and thus obtain the group message M.
Consider the group configuration as shown in the Figure 5. If the sender u_s wants to communicate the message M to the group members u_1, u_2 and u_3, then the sender chooses a random number r_0 and computes the ciphertext

$$(C_1^0, C_2^0) = (g^{r_0}, M.g^{GK_l+r_0+PK_0})$$

This ciphertext is then forwarded to its parent proxy P_1 which computes

$$\begin{aligned}(C_1^1, C_2^1) &= (C_1^0.g^{r_1}, C_2^0.g^{r_1+PK_1-PK_0}) \\ &= (g^{r_0+r_1}, M.g^{GK_l+r_0+r_1+PK_1})\end{aligned}$$

Proxy P_1 further forwards this ciphertext pair to its child proxies P_2 and P_3. Proxy P_2 computes

$$\begin{aligned}(C_1^2, C_2^2) &= (C_1^1.g^{r_2}, C_2^1.g^{r_2+PK_2-PK_1}) \\ &= (g^{r_0+r_1+r_2}, M.g^{GK_l+r_0+r_1+r_2+PK_2})\end{aligned}$$

and communicates this ciphertext pair to its child users u_1 and u_2. Each of these users can further compute M by evaluating

$$\frac{C_2^2}{C_1^2.g^{GK_l}.g^{PK_2}} = \frac{M.g^{GK_l+r_0+r_1+r_2+PK_2}}{g^{GK_l+r_0+r_1+r_2+PK_2}} = M$$

Similarly, proxy P_3 computes

$$\begin{aligned}(C_1^3, C_2^3) &= (C_1^1.g^{r_3}, C_2^1.g^{r_3+PK_3-PK_1}) \\ &= (g^{r_0+r_1+r_3}, M.g^{GK_l+r_0+r_1+r_3+PK_3})\end{aligned}$$

User u_3 can now compute

$$\frac{C_2^3}{C_1^3.g^{GK_l}.g^{PK_3}} = \frac{M.g^{GK_l+r_0+r_1+r_3+PK_3}}{g^{GK_l+r_0+r_1+r_3+PK_3}} = M$$

and obtain the group message M.

C. Limitation of the Scheme

The aforementioned scheme suffers from a crucial fault which renders the scheme unsafe for group communication. The very purpose of a group communication scheme is to strictly enable only the valid group members to get an access to the group messages and the group rekeying messages. However, in the scheme described above, the key g^{GK_l} is made a public key. This allows any proxy P_i which computes the ciphertext (C_1^i, C_2^i), to decrypt it using the public key g^{GK_l} and its own proxy key PK_i. The ciphertext computed by a proxy P_i is of the form

$$\begin{aligned}(C_1^i, C_2^i) &= (C_1^{par(P_i)}.g^{r_i}, C_2^{par(P_i)}.g^{r_i+PK_i-PK_{par(P_i)}}) \\ &= (g^{r_0+\ldots+r_i}, M.g^{GK_l+r_0+\ldots+r_i+PK_i})\end{aligned}$$

Note that the proxy P_i has access to the key PK_i as well as the public key g^{GK_l} and thus can compute

$$\frac{C_2^i}{C_1^i.g^{GK_l}.g^{PK_i}} = \frac{M.g^{GK_l+r_0+\ldots+r_i+PK_i}}{g^{GK_l+r_0+\ldots+r_i+PK_i}} = M$$

This causes the message M intended for the group members, to be accessible to the proxies, thus failing to satisfy the fundamental requirement of a group communication scheme.

IX. Security Analysis of Proxy Encryptions for Secure Multicast Key Management Scheme

Mukherjee et al. [37] proposed a framework for GKM using the concepts of PRE. The motivation behind the proposal was that in a multicast environment, the trust on the intermediate relaying nodes should be minimal for secure group communication. The rekeying information, though being forwarded by the intermediate nodes, should not be accessible to them. PRE is a technique which allows data forwarding by transformation in such a way that the intermediate proxy nodes do not learn any information from the ciphertext being transformed. Mukherjee et al. [37] mapped this concept of the proxy in a PRE scheme to the intermediate nodes in a multicast environment and proposed a framework for key management in multicast environment. Since the proposal by Mukherjee et al. [37] provides a framework and not the specific details of the group messages and their communication, we do not detail the specifics of the framework but only analyze the PRE scheme used in the scheme proposed by Mukherjee et al. [37].

A. PRE Scheme

In this Section, we describe the PRE scheme used by Mukherjee et al. [37] in the GKM framework and evaluate it based on the desirable properties of a PRE scheme. We consider a user U is identified by its public-private key pairs (pk_U, sk_U). A user X is considered to be the delegator in the scheme and the user Y is the delegatee. The PRE scheme follows the Elgamal encryption system where p is a prime such that Z_p^* is the group under which the scheme operates and g is the generator of Z_p^*.

1. Encryption:

$$(C_{X_1}, C_{X_2}) = (g^r, M.pk_X^r) = (g^r, M.g^{r.sk_X})$$

2. Decryption: To decrypt a ciphertext encrypted under the public key of X, the secret key of X is used as follows:

$$M = \frac{C_{X_2}}{(C_{X_1})^{sk_X}} = \frac{M.g^{r.sk_X}}{(g^r)^{sk_X}}$$

3. Re-encryption Key: The re-encryption is enabled by splitting the secret key sk_X used for decryption into two components sk_{1X} and sk_{2X} such that, $sk_X = sk_{1X} + sk_{2X}$. The first component sk_{1X} is provided to the proxy and the second component sk_{2X} is provided to the delegatee Y.

4. Re-encryption: A ciphertext (C_{X_1}, C_{X_2}) intended for the delegator X, is converted by the proxy as follows to facilitate the decryption by the delegatee:

$$\begin{aligned}(C'_{Y_1}, C'_{Y_2}) &= (C_{X_1}, \frac{C_{X_2}}{(C_{X_1})^{sk_{1X}}}) \\ &= (g^r, \frac{M.g^{r.sk_X}}{g^{r.sk_{1X}}}) \\ &= (g^r, M.g^{r.sk_{2X}})\end{aligned}$$

5. Decryption of re-encrypted ciphertext: A delegatee Y decrypts the re-encrypted ciphertext (C'_{Y_1}, C'_{Y_2}) by computing

$$\frac{C'_{Y_2}}{(C'_{Y_1})^{sk_{2X}}} = \frac{M.g^{r.sk_{2X}}}{(g^r)^{sk_{2X}}} = M$$

1) Properties of the PRE Scheme

1. Non-interactive: Computation of the re-encryption key material for a delegation from X to Y does not require the delegatee's secret key, rather it can only be computed from the delegator X's secret key sk_X.

2. Unidirectional: Possessing $rk_{X\to Y}$ allows the proxy to hold the first component sk_{1X} of the delegator X's secret key, whereas, the re-encryption keying material for $rk_{Y\to X}$ requires the proxy to possess sk_{1Y} which is not available to it via $rk_{X\to Y}$.

3. Not key optimal: The delegatee Y is required to store an additional key sk_{2X} to accept a delegation of decryption rights from X. The number of delegation keys required to be stored are directly proportional to the number of delegations accepted by a delegatee.

4. Not collusion safe: A collusion of the proxy and the delegatee provides them to have access to sk_{1X} and sk_{2X}, thus allowing for the computation of the secret key $sk_X = sk_{1X} + sk_{2X}$ of the delegator X.

5. Non-transitive: A proxy possessing its component of $rk_{X\to Y}$ and $rk_{Y\to Z}$ holds the keys sk_{1X} and sk_{1Y}. A re-encryption key from X to Z requires the proxy to hold one component of sk_X and the intended delegatee Z to hold the corresponding second component of sk_X. The proxy does not have an access to the secret key sk_X and thus cannot compute the re-encryption key components, one of which will is required to be held by the intended delegatee Z.

6. Transferable: A colluding proxy and the delegatee Y can compute the secret key sk_X of the delegator as described in the collusion susceptibility property and thus can redelegate the decryption rights to Z by splitting sk_X into two components, one of which is held by the proxy and one is communicated to Z.

7. Not temporary: The re-encryption enabled by the re-encryption keying material held by the proxy and the delegatee is such that it allows for re-encryption as long as the secret key of the delegator remains unchanged.

8. Proxy Visible: The decryption key sk_{2X} used by a delegatee Y for decryption of a re-encrypted ciphertext from delegatee X to it, is different from the decryption key sk_Y used by Y to decrypt a ciphertext directly intended for it.

9. Original access is allowed: Once a ciphertext of user X has been re-encrypted for a delegation using the sk_{1X} component by the proxy, the delegator X can himself decrypt the re-encrypted ciphertext by using the sk_{2X} component of sk_X, which it had computed to delegate the decryption rights.

X. Conclusion

We have shown the existing key management schemes based on proxy re-encryption schemes are not secure against the realistic strong active outsider adversary model. For practical applicability of these schemes, it is necessary for them to be secure against the strong active outsider adversary. Also, we have shown that the base proxy re-encryption scheme on which the construction of the key management schemes are based does not satisfy the crucial and desirable properties of the proxy re-encryption scheme which are necessary for the security. Designing the proxy re-encryption based key management scheme secure against the active outsider adversary and also satisfies the desirable properties of the key management schemes is a challenge.

Acknowledgment

This work is supported by Science and Engineering Research Board (SERB), Department of Science & Technology (DST), Government of India under Project No. ECR/2015/000428.

References

[1] X. Zou, B. Ramamurthy, S. S. Magliveras. *Secure Group Communications Over Data Networks*, Santa Clara, CA, USA: Springer-Verlag TELOS, 2004.

[2] S. Rafaeli, D. Hutchison. A survey of key management for secure group communication, *ACM Computing Surveys*, 35 (3), pp. 309–329, Sep. 2003.

[3] S. Xu. On the security of group communication schemes. *Journal of Computer Security*, 15 (1), pp. 129–169, Jan. 2007.

[4] S. Xu. On the security of group communication schemes based on symmetric key cryptosystems. In *Proceedings of the 3rd ACM Workshop on Security of Ad Hoc and Sensor Networks*, New York, USA, pp. 22–31, 2005.

[5] D. Manz, J. Alves-Foss, S. Zheng. Network simulation of group key management protocols. *Journal of Information Assurance and Security*, 3 (1), pp. 67–79, 2008.

[6] C. K. Wong, M. Gouda, S. S. Lam. Secure group communications using key graphs, *IEEE/ACM Transactions on Networking*, 8 (1), pp. 16–30, Feb 2000.

[7] O. Cheikhrouhou. Secure group communication in wireless sensor networks: A survey, *Journal of Network and Computer Applications*, 61, pp. 115 – 132, 2016.

[8] Y. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu, M. Galloway. A survey of key management schemes in wireless sensor networks, *Computer Communications*, 30 (11), pp. 2314 – 2341, 2007.

[9] M. A. Simplcio, P. S. Barreto, C. B. Margi, T. C. Carvalho. A survey on key management mechanisms for distributed wireless sensor networks, *Computer Networks*, 54 (15), pp. 2591 – 2612, 2010.

[10] R. Aparna, B. B. Amberker. A key management scheme for secure group communication using binomial key trees, *International Journal of Network Management*, 20 (6), pp. 383–418, Nov. 2010.

[11] A. T. Sherman, D. A. McGrew. Key establishment in large dynamic groups using one-way function trees, *IEEE Transactions on Software Engineering*, 29 (5), pp. 444–458, May 2003.

[12] B. R. Purushothama, N. Koti. Security analysis of tree and non-tree based group key management schemes under strong active outsider attack model. In *4th International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015*, Kochi, India, August 10-13, 2015, pp. 1825–1829, 2015.

[13] J. Zhou, Y. Ou. Key tree and chinese remainder theorem based group key distrubution scheme, *Journal of the Chinese Institute of Engineers*, 32 (7), pp. 967–974, 2009.

[14] X. Zheng, C. T. Huang, M. Matthews. Chinese remainder theorem based group key management. In *Proceedings of the 45th Annual Southeast Regional Conference (ACM-SE 45)*, New York, USA: ACM, pp. 266–271, 2007.

[15] J. Zhou, Y.h. Ou. Key tree and chinese remainder theorem based group key distribution scheme. In *9th International Conference on Algorithms and Architectures for Parallel Processing, ICA3PP 2009*, Springer Berlin Heidelberg, pp. 254–265, 2009.

[16] M. Y. Joshi, R. S. Bichkar. Scalable key transport protocol using chinese remainder theorem. In *International Symposium on Security in Computing and Communications, SSCC 2013*, Mysore, India, pp. 397–402, 2013.

[17] G. H. Chiou, W. T. Chen. Secure broadcasting using the secure lock, *IEEE Transactions on Software Engineering*, 15 (8), pp. 929–934, Aug 1989.

[18] P. Vijayakumar, S. Bose, and A. Kannan. Chinese remainder theorem based centralised group key management for secure multicast communication, *IET Information Security*, 8 (3), pp. 179–187, May 2014.

[19] B. R. Purushothama, A. P. Verma, A. Kumar. Security analysis of key management schemes based on chinese remainder theorem under strong active outsider adversary model. In *Proceedings of the 5th International Symposium on Security in Computing and Communications, SSCC 2017*, Manipal, India, September 13-16, 2017, pp. 215–225, 2017.

[20] M. Eltoweissy, M. H. Heydari, L. Morales, I. H. Sudborough. Combinatorial optimization of group key management, *Journal of Network and Systems Management*, 12 (1), pp. 33–50, Mar 2004.

[21] D. Naor, M. Naor, J. B. Lotspiech. Revocation and tracing schemes for stateless receivers. In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, London, UK: Springer-Verlag, pp. 41–62, 2001.

[22] Q. A. Al-Haija. Toward secure non-deterministic distributed wireless sensor network using probabilistic key management approaches. *Journal of Information Assurance and Security*, 6 (1), pp. 010–018, 2011.

[23] S. Athmani, A. Bilami, D. E. Boubiche. Edak: An efficient dynamic authentication and key management mechanism for heterogeneous wsns. *Future Generation Computer Systems*, 92, pp. 789–799, 2019.

[24] C. Esposito, M. Ficco, A. Castiglione, F. Palmieri, A. De Santis. Distributed group key management for event notification confidentiality among sensors. *IEEE Transactions on Dependable and Secure Computing*, 2018.

[25] M. Ali, B. M. Salim. An efficient group key management using clustering algorithm for mobile ad hoc networks. In *Third International Congress on Information and Communication Technology*, Springer, pp. 107–118, 2019.

[26] K. Selvakumar, N. Seethalakshmi. Secure group key management protocol for mobile ad hoc networks. *Cluster Computing*, pp. 1–7, 2018.

[27] V. Janani, M. Manikandan. A genetic-based bayesian framework for stateless group key management in mobile ad hoc networks. In *Soft Computing for Problem Solving*, Springer, pp.425–434, 2019.

[28] M. Saha, D. R. Chowdhury. A conference key agreement protocol for mobile environment. *Journal of Information Assurance and Security*, 4, pp. 60–68, 2009.

[29] B. R. Purushothama, A. P. Verma. Security analysis of group key management schemes of wireless sensor network under active outsider adversary model. In *6th International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, Udupi (Near Mangalore), India, September 13-16, 2017, pp. 988–994, 2017.

[30] A. Chaudhari, G. Pareek, B. R. Purushothama. Security analysis of centralized group key management schemes for wireless sensor networks under strong active outsider adversary model. In *6th International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, Udupi (Near Mangalore), India, September 13-16, 2017, pp. 1576–1581, 2017.

[31] J. Hur, Y. Shin, H.Yoon. Decentralized group key management for dynamic networks using proxy cryptography. In *Proceedings of the 3rd ACM Workshop on QoS and Security for Wireless and Mobile Networks, (Q2SWinet '07)*, New York, NY, USA: ACM, pp. 123–129, 2007.

[32] Y. Chen, J. D. Tygar, W. Tzeng. Secure group key management using uni-directional proxy re-encryption schemes. In *30th IEEE International Conference on Computer Communications, INFOCOM 2011*, pp. 1952–1960, 10-15 April 2011, Shanghai, China, 2011.

[33] Y. Han, X. Gui, X. Wu, and X. Yang. Proxy encryption based secure multicast in wireless mesh networks. *J. Netw. Comput. Appl.*, 34 (2), pp. 469–477, Mar. 2011.

[34] C. Y. Huang, Y. P. Chiu, K. T. Chen, C.L. Lei. Secure multicast in dynamic environments. *Computer Networks*, 51 (10), pp. 2805–2817, Jul. 2007.

[35] Z. Wang, X. Du, Y. Sun. Group key management scheme based on proxy re-cryptography for near-space network. In *2011 International Conference on Network Computing and Information Security (NCIS)*, 1, IEEE, pp. 52–56, 2011.

[36] G. Ateniese, K. Fu, M. Green, S. Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage, *ACM Transactions on Information and System Security (TISSEC'06)*, 9 (1), pp. 1–30, 2006.

[37] R. Mukherjee, J. W. Atwood. Proxy encryptions for secure multicast key management. In *28th Annual IEEE International Conference on Local Computer Networks, 2003. LCN '03. Proceedings.*, pp. 377–384, Oct 2003.

[38] G. Pareek, B. R. Purushothama. Provably secure group key management scheme based on proxy re-encryption with constant public bulletin size and key derivation time, *Sādhanā*, 43 (9), p. 137, Jul 2018.

[39] Z. Wang, X. Du, Y. Sun. Group key management scheme based on proxy re-cryptography for near-space network. In *2011 International Conference on Network Computing and Information Security*, 1, pp. 52–56, May 2011.

[40] Z. Minqing, W. Xuguang, H. Yiliang, Y. Xiaoyuan. Group key management using proxy re-encryption for dynamic networks. In *2010 Second International Workshop on Education Technology and Computer Science*, 1, pp. 190–193, March 2010.

[41] Y. Shin and J. Hur. Scalable and efficient approach for secure group communication using proxy cryptography, *Wireless Networks*, 18 (4), pp. 413–425, 2012.

[42] Y. P. Chiu, C. L. Lei, C. Y. Huang. Secure multicast using proxy encryption. In *Information and Communications Security*, S. Qing, W. Mao, J. López, and G. Wang, (eds.), Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 280–290, 2005.

[43] M. Blaze, G. Bleumer, M. Strauss. Divertible protocols and atomic proxy cryptography. In *EUROCRYPT*, Springer-Verlag, pp. 127–144, 1998.

[44] M. Jakobsson. On quorum controlled asymmetric proxy re-encryption. In *Public Key Cryptography*, Berlin, Heidelberg, Springer Berlin Heidelberg, pp. 112–121, 1999.

[45] L. Zhou, M. A. Marsh, F. B. Schneider, A. Redz. Distributed blinding for distributed elgamal re-encryption. In *25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, pp. 824–824, June 2005.

[46] A. Ivan, Y. Dodis. Proxy cryptography revisited. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2003.

Author Biographies

Shravani Mahesh Patil is a Postgraduate student at the Department of Computer Science and Engineering, National Institute of Technology Goa, India. She works in the area of Information Security and Cryptography, Security Analytics.

Purushothama B R obtained his Ph.D in Computer Science and Engineering from National Institute of Technology Warangal (NIT Warangal),India and did his M.Tech in Computer Science and Engineering from National Institute of Technology Karantaka (NIT) Surathkal, India. He is currently working as Assistant Professor in the Department of Computer Science and Engineering at National Institute of Technology Goa, India. His areas of interest are Cryptography, Cloud Security, Data Security, Provable Security, Network Security and Algorithms.