A Dynamic Threshold Decryption Scheme Using Bilinear Pairings

Brian King
Department of Electrical and Computer Engineering, Indiana University - Purdue University Indianapolis
723 West Michigan Street, SL160, Indianapolis, IN 46202, USA
(Email: briaking@gmail.com)

(Received May. 15, 2013; revised and accepted Jan 13 & Feb. 6, 2014)

Abstract

A dynamic threshold sharing scheme is one that allows the set of participants to expand and contract. In this work we discuss dynamic threshold decryption schemes using bilinear pairing. We discuss and analyze existing schemes, demonstrate an attack and construct a significantly more efficient secure scheme.

Keywords: Bilinear pairing, dynamic threshold scheme, ECC

1 Introduction

Secret sharing is a mechanism that is used to share out a secret to multiple parties such that only those authorized sets are allowed to recover the secret key. Threshold secret sharing is an example of a secret sharing scheme, where the authorized sets consists of those groups of participants whose membership is greater than or equal to the threshold. A t out of n threshold sharing [2, 10] is such that any set of participants that contains t or more are authorized and can recover the secret. The most common use of threshold secret sharing is to build threshold cryptographic applications. Threshold cryptography refers to a technique where threshold secret sharing is used to compute a function of the secret rather than the secret itself. Examples of functions/applications would include a decryption of ciphertext and signature schemes. Threshold cryptography has been used to describe many group oriented applications [6].

Today, it is common security technique that is used to achieve computationally secure group access. A dynamic threshold sharing scheme [8] is threshold sharing scheme where the participant set is dynamic, allowing it to expand, as well as contract. Identity based encryption is a technique such that some public identity information is used as a public key. Identity based encryption was first proposed by Shamir [11]. In [3], Boneh and Franklin constructed an identity based encryption scheme based on bilinear pairings.

In this paper we discuss a dynamic threshold encryption scheme, we discuss two current schemes and discuss attacks in their schemes. We then provide an improved dynamic threshold decryption scheme. We assume we have the following system. Users enroll in a encryption/decryption scheme. Once enrolled their identification ID is registered. The service of the system is such that users can register their identity to a trusted third party, denoted by TTP, who then constructs and publishes their identity-based-public-key. The precise process of the registration is outside the scope of our work.

The user can then have messages encrypted to them based on the system public key and their identity and then have threshold servers decrypt the ciphertext for the user. The process in which the user makes the request for a decryption is outside the scope of the paper. The servers are dynamic in nature and can grow and contract over time.

1.1 Dynamic Threshold Decryption Scheme

The concept of a *dynamic threshold decryption scheme* [8], is such that a public key encryption scheme exists, the decryption key `sk` is shared out to a set of n decryption servers, denoted by $\Gamma_1, \Gamma_2, \ldots, \Gamma_n$, in such a way that any t out of n can decrypt a message which is encrypted using the public key `pk`, and the membership of these servers is dynamic in nature.

The goals for a dynamic dynamic sharing scheme [8] are:

- The system can refresh the decryption key without having to modify any of the shares of the decryption servers.
- If the system adds a new decryption server then the systems does not have to modify the decryption key

nor do they have to modify any of the other decryption server's shares.

- The removal of a decryption server does not require the modification of any other decryption server's shares, further it does not require the modification of the decryption key.

- A decryption server can refresh its private key without requiring any other decryption server to modify their private key, further this refresh does not require the system to modify of any other decryption server shares.

In order to develop an enhanced system we propose a revised set of criteria, to support these revised features, additional public information will be available in a publicly available setting, such as a bulletin board. This additional bulletin board information is typically used to support a set of authorized servers ability to reconstruct the decryption key and/or to decrypt a message for a user.

In [5], Chen, Gollman, Mitchell and Wild introduced the concept of reusable polynomials for secret sharing with a goal of supporting dynamic thresholds. The main properties that they were interested in were:

Perfect security or computational security. A secret sharing scheme is perfectly secure if unauthorized subsets of shareholders cannot obtain information about the secret. A scheme is computationally secure provided it is computationally infeasible to determine the secret from an unauthorized subset.

Verifiability. First, each shareholder should be able to verify their received share to detect a dishonest or faulty dealer. Secondly, during secret reconstruction a forged share contributed by a cheating shareholder can be detected by the other shareholders.

Online shareholders. Shareholders can dynamically join or leave the sharing group without having to redistribute new shares secretly to the existing shareholders.

Reusable shares. Shares need to be reusable even after the shared secret has been reconstructed.

Our criteria for secure dynamic threshold sharing:

- The system can refresh decryption key without having to contact and/or send new shares to any of the decryption servers.

- When the system adds a new decryption server then they do not have to modify the decryption key nor do they have contact and/or send new shares to any of the other decryption server's shares.

- The removal of a decryption server does not require contact and/or the sending new shares of any other decryption server's shares, further it does not require the modification of the decryption key.

- A decryption server can refresh its private key without requiring the any other decryption server to modify their private key, further it does not require contact and/or sending new shares to any other decryption server shares.

- Each server should be able to verify their shares compute the secret.

- All system/shareholder (server) computations are efficient.

This set of criteria allows for the use of public setting modification (for example, shares placed on a public bulletin board).

1.2 Bilinear Pairings

A mathematical tool that we utilize in our work will be the bilinear pairing. Let $\mathbb{G}$, and $\mathbb{G}_1$ be cyclic groups of prime order p, such that both $\mathbb{G}$ and $\mathbb{G}_1$ are multiplicative groups.

Definition 1. *A map $e : \mathbb{G}\times\mathbb{G} \to \mathbb{G}_1$ is said to be bilinear pairing if it has the following properties:*

Bilinearity. $e(g^a, w^b) = e(g, w)^{ab}$ *for all* $g, w \in \mathbb{G}$ *and* $a, b \in \mathbb{Z}_p^*$.

Non-degeneracy. $e(g, g) \neq 1$ *in* $\mathbb{G}_1$ *and* $g \neq 1$ *in* $\mathbb{G}$.

Computability. *There exists an efficient algorithm that computes* $e(g, w)$ *for all* $g, w \in \mathbb{G}$.

We will assume that the *discrete log problem* (DLOG) is hard. The DLOG problem is such that given group $\mathbb{G}$, and $g \in \mathbb{G}$ and g^k it is s "computationally infeasible" to determine k. We will assume that the discrete log problem is "hard" in $\mathbb{G}_1$.

In this paper we will be using a bilinear map e. We assume that the discrete log problem is "hard" even in the presence of the bilinear map, that is, given generator g, the value g^a and the pairing map e, it is "hard" to compute the exponent a. We will assume that *Computational Diffie-Hellman (CDH)* problem is also 'hard" in the presence of the bilinear map, thus given g^a, g^b, and $e(g^a, g^b)$ it is hard to compute ab. We will assume that the CDH problem is hard in $\mathbb{G}_1$. The *Decision Diffie-Hellman (DDH)* problem is the problem concerning whether one can distinguish between (g, g^a, g^b, g^c) and (g, g^a, g^b, g^{ab}). The *Decision Diffie-Hellman (DDH)* problem is "easy" in $\mathbb{G}_1$ due to the existence of the bilinear map e. Consequently, we will be working in an algebraic setting described by Boneh et al. [4] as the *Gap Diffie-Hellman*

(GDH) group. The group $\mathbb{G}_1$ is called a GDH group if DDH is easy in $\mathbb{G}_1$ but CDH is hard. Thus $\mathbb{G}_1$ is a GDH group.

1.3 Identity-based Encryption

The concept of identity-based encryption was proposed by Shamir in 1984 [11]. The construction of an identity-based encryption scheme was an open problem until solved by Boneh and Franklin in [3]. Today there are a number of identity based encryption schemes proposed, we refer the reader to a survey of the schemes [1].

2 Construction of Dynamic Threshold Decryption Scheme from Pairing

In this section we discuss a dynamic threshold decryption scheme proposed by Long and Chen [9]. Unfortunately there is a typographical error in their work, thus placing a level of ambiguity to their scheme. In [7] Kim, Lim, Yie, and Kim analyzed the Long scheme, because of the typographical error they had to make an interpretation of the error, their interpreted scheme was flawed. In Kim et al.'s cryptanalysis, they showed that their interpretation of the Long et al. scheme is insecure. Long and Chen constructed their *dynamic threshold decryption scheme* using bilinear pairings. Their scheme attempts to solve the problem of decrypting the ciphertext without compromising the master key, and was inspired by [12]. The Long et al. scheme is summarized in the following steps:

Setup. There is a trusted party private key generator (TTP) which chooses two bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$, where each group has prime order p. Let g be a generator of $\mathbb{G}_1$, and $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ a bilinear map. The messages are denoted by M. Assume all messages M belong to $\mathbb{G}_1$. We assume that each user u has a public key ID_u that belongs to $\mathbb{Z}_p^*$. The TTP selects random $x, y \in \mathbb{Z}_p^*$ and computes $X = g^x, Y = g^y$. The public parameters are denoted by $cp = (g, X, Y)$ and the master key $mkey = (x, y)$ where x remains secret and y is secret but renewed periodically.

KeyGen. Initially there are n decryption servers $\Gamma_1, \Gamma_2, ..., \Gamma_n$. Each server Γ_i possesses a secret key $s_i \in \mathbb{Z}_p^*$ and a corresponding public key $P_i = g^{s_i}$. The TTP selects a random polynomial $f(z)$ of degree $t-1$ over $\mathbb{Z}_p^*$ by selecting $b_1, b_2, \ldots b_{t-1}$ from $\mathbb{Z}_p^*$, the polynomial $f(z)$ satisfies $f(z) = y + \sum_{i=1}^{k-1} b_i z^i$, here $b_{k-1} \in \mathbb{Z}_p^*$. For each i, the TTP computes

$$k_i = g^{\frac{f(i)}{(ID+x)P_i^y}}$$

and $v_i = e(g, g)^{f(i)}$. The TTP publishes k_i and v_i on the public available site (which we will call the bulletin board).

Encryption. Suppose *Alice* would like to transmit message M to *Bob* privately. She gets *Bob*'s identification, denoted by ID, as well as the *master TTP keys* X and Y. She then encrypts message M with public key ID, by picking up a random $S \in \mathbb{Z}_p^*$ and computes the ciphertext C by $C = (g^{S \cdot ID} \cdot X^S, e(g, Y)^S \cdot M) = (A, B)$.

Γ_i 's Sub-decryption. After Bob receives the decrypted message that was sent by Alice, Bob can ask the servers to decrypt it. This can be achieved whenever t decryption servers $\Gamma_{i_1}, \ldots, \Gamma_{i_t}$ cooperate and reconstruct the message by utilizing their shares (on the bulletin board) of the decryption key in the t of n threshold sharing scheme. In this step, we illustrate how a server Γ_i calculates its decryption share δ_i of the ciphertext, which is computed with the use of the server's private key s_i. According to [9], the decryption server Γ_i calculates the share δ_i by computing

$$\delta_i = e(A, g^{Y^{s_i}})^{k_i} = \cdots = e(g, g)^{S \cdot f(i)}. \qquad (1)$$

Note: *The formula given in Equation (1) is wrong!* We address this issue in Section 2.1.

Decryption. Assuming decryption servers $\Gamma_1, \Gamma_2, ..., \Gamma_t$ want to decrypt the ciphertext, each server Γ_i computes δ_i and sends it to the combiner who computes Δ by:

$$\begin{aligned}
\Delta &= \prod_{j=1}^{t} (\delta_j)^{\prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{-i}{j-i}} \\
&= \prod_{j=1}^{t} e(g, g)^{sf(j) \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{-i}{j-i}} \\
&= e(g, g)^{\sum_{j=1}^{t} sf(j) \prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{-i}{j-i}} \\
&= e(g, g)^{sy} \\
&= e(g, Y)^s.
\end{aligned}$$

Note: *The above calculation only make sense provided that* $\delta_j = e(g, g)^{S \cdot f(j)}$. Again this is addressed in Section 2.1.

If Δ equals $e(g, Y)^S$, then M can be recovered by computing $M = B \cdot \Delta^{-1}$. The use of the bulletin board allows Long et al. to achieve the revised dynamic properties. The share v_i can be used by each server Γ_i to verify the

correctness of the share k_i, a property we are interested in satisfying. Further any server Γ_j can verify the correctness of the shares $v_1, \ldots v_n$ by selecting any t of them and computing

$$e(g,g)^y \stackrel{?}{=} \prod_{i=1}^{t} v_{w_i}^{\frac{\prod_{i=1, i\neq j}^{t} -w_j}{w_i - w_j}}.$$

Long et al. claimed that their scheme satisfied the following dynamic threshold requirements.

TTP refreshes secret key. In the case a new secret key y_{new} is selected, then $Y_{new} = g^{y_{new}}$ is computed and a new polynomial f_{new} is selected, new shares $k_{i,new}$ and $v_{i,new}$ will be shared out to the servers $\Gamma_1, \ldots, \Gamma_n$.

TTP adds new decryption server. In the case the TTP adds a new server Γ_{n+1}, they simply use the polynomial value $f(n+1)$ and generate a new share

$$k_{n+1} = g^{\frac{f(n+1)}{(ID+x)P_{n+1}^y}} \text{ and } v_{n+1} = e(g,g)^{f(n+1)}$$

here P_{n+1} is Γ_{n+1}'s public key.

TTP removes a decryption server. Assume without loss of generality that server Γ_n is dismissed then a new polynomial f_{new} is selected with the same secret key y and new shares $k_{i,new}$ and $v_{i,new}$ will be shared out to the bulletin board by the TTP for the servers $\Gamma_1, \ldots, \Gamma_{n-1}$.

Server Γ_i refreshes their secret key s_i. If Γ_i refreshes their secret key and select $s_{i,new}$ then they will compute $P_{i,new} = g^{s_{i,new}}$ and new shares $k_{i,new}$ and $v_{i,new}$ will be shared out to the bulletin board by the TTP for the servers.

Remarks: Clearly this scheme does not possess the security (nor correctness) that the authors claim. More importantly, this is very inefficient. In reality the labelling of shares as k_i and v_i is inaccurate as they depend not only on the server Γ_i but also on the user's identification ID. That is, if there are m users $\{ID_1, ID_2, \ldots, ID_m\}$ then there are m distinct (k_i, v_i) pairs (as illustrated below–one pair for each user ID). That is, we have

	Γ_1	Γ_2	$\cdots$	Γ_n
ID_1	(k_{1,ID_1}, v_{1,ID_1})	(k_{2,ID_1}, v_{2,ID_1})	$\cdots$	(k_{n,ID_1}, v_{n,ID_1})
ID_2	(k_{1,ID_2}, v_{1,ID_2})	(k_{2,ID_2}, v_{2,ID_2})	$\cdots$	(k_{n,ID_2}, v_{n,ID_2})
$\vdots$	$\vdots$		$\ddots$	
ID_m	(k_{1,ID_m}, v_{1,ID_m})	(k_{2,ID_m}, v_{2,ID_m})	$\cdots$	(k_{n,ID_m}, v_{n,ID_m})

Thus the cost of executing the refresh properties are (in big O notation) is described in Table 1.

Table 1: Computational cost of Long et al. scheme

operation	computational cost
TTP refreshes secret key	$O(mn)$
TTP adds new decryption server	$O(m)$
TTP removes a decryption server	$O(mn)$
server Γ_i refreshes their secret key	$O(m)$

2.1 Kim et al.'s Interpretation of the Long Scheme

Clearly there is a typographical error in Long et al. scheme. In [7] Kim, Lim, Yie, and Kim cryptanalyzed the Long et al. scheme. Unfortunately due to the typographical error in the Long et al. paper [9], Kim et al. [7] had to interpret the scheme, they interpreted the Long scheme as follows:

Setup. *Same as before.*

KeyGen. *Same as before*, exception: for each i, the TTP computes $k_i = g^{\frac{f(i)}{(ID+x)p_i^y}}$, $v_i = e(g,g)^{f(i)}$ and publishes k_i, v_i.

Encryption. *Same as before.*

Γ_i 's Sub-decryption. Bob can receive the message sent by Alice by having t servers $\Gamma_{i_1}, \ldots, \Gamma_{i_t}$ reconstruct the message by utilizing their shares of the decryption key in the t of n threshold sharing scheme. In this step, the server Γ_i calculates its decryption share δ_i of the ciphertext as follows:

$$\begin{aligned} \delta_i &= e(A, k_i \cdot Y^{s_i}) \\ &= e(g,g)^{S\cdot(ID+x)\cdot\frac{f(i)}{ID+x}} \\ &= e(g,g)^{S\cdot f(i)}. \end{aligned}$$

The above derivations are correct.

Decryption. Assuming t decryption servers $\Gamma_1, \Gamma_2, ..., \Gamma_t$ wish to decrypt the ciphertext for user ID, one of the servers collects $\delta_1, \delta_2, ..., \delta_t$ and computes Δ as follows:

$$\begin{aligned} \Delta &= \prod_{j=1}^{t} \delta_j^{\prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{-i}{j-i}} \\ &= \prod_{j=1}^{t} e(g,g)^{sf(j)\prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{-i}{j-i}} \\ &= e(g,g)^{\sum_{j=1}^{t} sf(j)\prod_{\substack{i=1 \\ i\neq j}}^{t} \frac{-i}{j-i}} \\ &= e(g,g)^{sy} \\ &= e(g,Y)^s. \end{aligned}$$

Each server Γ_j sends their δ_j to Bob who then computes Δ. Lastly Bob computes $M = B \cdot \Delta^{-1}$, which is the final step of decryption.

The dynamic properties that need to be supported (in the Kim et al. interpretation) are as follows:

TTP refreshes secret key. *Same as before*

TTP adds new decryption server. *Same as before.*

TTP removes a decryption server. *Same as before.*

server Γ_i refreshes secret key s_i. *Same as before.*

In their work [7], Kim et al. successfully attacked their interpretation of the Long scheme. We find that Kim et al. misinterpreted the Long scheme [9], which we describe in Section 2.3.

2.2 Kim et al.'s Attack of Their Interpreted Long Scheme

In [7] the authors attacked their interpreted version of the Long scheme. The attack they constructed was such that it violated the decryption requirement that only t authorized servers can decrypt an encrypted message for any party. The Kim et al. attack can be summarized as follows: Suppose server Γ_w is malicious. They claim want an update of their public key P_w but rather than selecting a new secret key, suppose they wish to attack server Γ_1. They use Γ_1's public key P_1 and select $r \in \mathbb{Z}_p^*$ and compute P_1^r and sends this to the TTP claiming that P_1^r is their "new public key", calling it $P_{w,new}$. Thus $P_{w,new} = P_1^r$. The TTP not knowing that server Γ_w has misrepresented their new public key, refreshes Γ_w's shares $k_{w,new}$ and $v_{w,new}$. Here $k^{w,new} = g^{\frac{f(w)}{ID+x}} P_1^{ry}$. Because server Γ_w can compute $g^{\frac{w}{ID+x}}$ the server Γ_w now knows P_1^{ry} since the server knows r it can compute P_1^y by computing $(P_1^{ry})^{r^{-1}}$. Then using k_1 it computes $g^{\frac{f(1)}{ID+x}}$ by computing $k_1 \cdot (P_1^y)^{-1}$. Now Γ_w knows $g^{\frac{f(1)}{ID+x}}$ and $g^{\frac{f(w)}{ID+x}}$. Now together with $t-2$ other servers it can compute $g^{\frac{y}{ID+x}}$ which is $Y^{\frac{1}{ID+x}}$. Then given ciphertext C, the $t-1$ servers can compute $e(g^{S \cdot ID} \cdot X^S, Y^{\frac{1}{ID+x}}) = e(g,g)^{Sy}$, denote this by Δ. Then $M = C \cdot \Delta^{-1}$. Hence Γ_w has successfully defeated the threshold requirement since a coalition of $t-1$ servers can decrypt messages.

Note: The server Γ_w could actually complete this attack $t-1$ times and be able to decrypt by itself. Though $t-1$ refreshes may make the TTP suspicious of their behavior.

The Kim et al. attack has successfully defeated the threshold requirement. Kim argued that the only way to prevent such attack is that the system has to renew the secret shares of all decryption servers whenever one of the decryption server renews its secret key, however this is problematic in that there are n servers and m users as described in Table 1. Thus the cost is $O(mn)$, which is too much. We solved this problem constructing a more efficient scheme, we will discuss a new attack, which is relevant to all version of the Long scheme. However, we will first demonstrate that the Kim et al. interpretation of the Long scheme was incorrect.

2.3 Our Interpretation of the Long Scheme

The interpretation of the Long [9] scheme by Kim et al. [7] led to the attach described in Section 2.2, that is by their interpretation they were able to construct the attack. After observing the Long scheme [9], it became apparent to us that the error was merely in the presentation of Equation (1). That is the share k_i was expressed in the Long scheme as

$$k_i = g^{\frac{f(i)}{(ID+x)P_i^y}}. \tag{2}$$

But clearly based on the assumption that Equation (2) is correct then the δ_i (decryption subshare) is incorrect. We then observed that based on the assumption Equation (2) is correct that δ_i should be calculated as $\delta_i = e(A, k_i)^{Y^{s_i}}$ because

$$\begin{aligned} \delta_i &= e(A,k_i)^{Y^{s_i}} \\ &= e(A,g)^{Y^{s_i} \cdot \frac{f(i)}{(ID+x)P_i^y}} \\ &= e(g,g)^{S \cdot (ID+x) \cdot (P_i)^y \cdot \frac{f(i)}{(ID+x)P_i^y}} \\ &= e(g,g)^{S \cdot f(i)}, \end{aligned}$$

which is exactly what the Long scheme required. Then if a threshold of t servers (say $\Gamma_1, \ldots, \Gamma_t$) need to compute a function of the secret

$$\Delta = \prod_{j=1}^{t} \delta_j^{\prod_{\substack{i=1 \\ i \neq j}}^{t} \frac{-i}{j-i}} = e(g,Y)^s$$

The irony is that this version is not susceptible to the attack described by Kim et al. [7], and that the attempt to fix the typographical error in the Long scheme introduced the security weakness that allowed them to attack it. At the same time we note an attack on the Long scheme (both versions), as well as Kim's suggested fix.

3 Our Attack

Thus we see the Kim et al. attack was due to their interpretation of the Long scheme. Suppose Alice encrypts message M to user denoted by ID_0 then ciphertext C satisfies $C = (g^{S \cdot ID_0} \cdot X^S, e(g,Y)^S \cdot M) = (A, B)$. Now consider the following attack. Suppose a server leaves the network, in particular suppose the server is removed and is intent on causing problems to the network that removed them. We assume without loss of generality that

it is server Γ_n. Now with the removal of this server the threshold is a t out of $n-1$ and there are only $n-1$ players, all other parties are considered outsiders and outsider help does not contribute towards the threshold. For example, as an extreme, if we have t helpers who are not authentic decryption servers but for some reason posses valid shares in a t out od X scheme then they should not be able to decrypt. Suppose Γ_n gives their information $g^{\frac{f(n)}{ID+x}}$ to $t-1$ parties. The server Γ_n is now considered as an outsider and doesn't count towards the threshold, thus it should not be be considered as help to achieve a threshold that decrypt a message intended for a user. No matter which approach is taken (the original Long scheme or the Kim interpreted scheme or our interpreted scheme) all k_i and v_i are reshared out because of the dismissal of Γ_n. However if Γ_n provides the information $g^{\frac{f(n)}{ID+x}}$ to a set of $t-1$ participants, for example $\{\Gamma_1, \Gamma_2, \ldots, \Gamma_{t-1}\}$, let us call this set $\mathcal{ADV}$, then $\mathcal{ADV}$ will be able to decrypt. This is because the participants in $\mathcal{ADV}$ can use their old shares $k_{1,old}, \ldots, k_{t-1,old}$ [1]. If Γ_n sends $g^{\frac{f(n)}{ID_0+x}}$ to this group $\mathcal{ADV}$, then $\mathcal{ADV}$ can decrypt the message and $\mathcal{ADV}$ contains only $t-1$ authentic members, but this violates that no threshold less than t authentic members can decrypt. Note that any dismissed party can give their share to other members, allowing an unauthorized set to decrypt (below threshold). In fact it is possible after $t-1$ parties are dismissed that a single party could be given all shares and thus they could decrypt by themselves.

4 Our Protocol

4.1 The Protocol

Our goal is to create an efficient dynamic threshold scheme based on bilinear pairings.

Setup. Two multiplicative groups $\mathbb{G}$ and $\mathbb{G}_1$ of order prime p are selected such that there exist a bilinear map $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_1$. The TTP selects $g \in \mathbb{G}$ where $g \neq 0$.

Key Generation. The TTP selects a secret key denoted by y and compute the corresponding public key $Y = g^y$. The TTP selects two temporal keys x_1 and x_2 and computes $V = g^{x_2}$ and $W = g^{x_2 x_1}$. The values Y, V and W are posted on a publicly available web site, such as a bulletin board.

Each server Γ_i selects a secret key w_i and computes their public key $P_i = g^{w_i}$. They publish their public key. The TTP will keep a local copy of the server's public keys.

[1] Though the bulletin board has been updated with new shares the shareholders may have prestored the older shares.

Share Generation. The TTP selects a random polynomial $f(z) = \sum_{i=1}^{t-1} b_i \cdot z^i$ of degree $t-1$ such that $f(0) = x_2^{-1} y \mod p$, i. e. $b_0 = x_2^{-1} \cdot y$. We assume initially there are n decryption servers $\{\Gamma_1, \ldots, \Gamma_n\}$ then the TTP computes the share $k_i = f(i) \cdot P_i^y$ and $v_i = e(V, g)^{f(i)}$. The TTP publishes (k_i, v_i) on bulletin board.

User Registration of Their Identity ID. Suppose a user wishes to register their identity ID_0 with the TTP. They interact with the TTP in a communication that establishes that ID_0 is their identity (this communication to achieve this is outside our scope). Once this is established the TTP publishes Z_{ID_0} where $Z_{ID_0} = g^{\frac{1}{ID_0+x_1}}$ onto the public site (bulletin board).

Encryption of Message M to User with Identity ID_0. Suppose *Alice* would like to transmit message M to *Bob* privately. She gets *Bob*'s identification, denoted by ID_0, as well as the TTP's public key Y and the two temporal key X. She then encrypts message M with public key ID_0, by picking up a random $S \in \mathbb{Z}_p^*$ and computes the ciphertext C by $C = (W^S \cdot V^{S \cdot ID_0}, e(g, Y)^S \cdot M) = (A, B)$.

Generation of Decryption Server Γ_i Decryption Shares. User Bob with identity ID_0 requests to the decryption servers that the ciphertext $C = (A, B)$ be decrypted. Assuming t servers respond, say $\{\Gamma_{i_1}, \ldots, \Gamma_{i_t}\}$, each of these servers will compute a decryption share based on the ciphertext and their share of the decryption key. For each i_r, server Γ_{i_r} computes the decryption share δ_{i_r} where δ_{i_r} satisfies

$$\begin{aligned} \delta_{i_r} &= e(A^{k_{i_r} \cdot Y^{-w_{i_r}}}, Z_{ID_0}) \\ &= e(A^{f(i_r)}, Z_{ID_0}) \\ &= e(g^{(x_2 x_1 S + x_2 S I D_0) f(i_r)}, g^{\frac{1}{ID_0+x_1}}) \\ &= e(g, g)^{S x_2 \cdot f(i_r)}. \end{aligned}$$

Here w_{i_r} is the server Γ_{i_r}'s secret key.

Decryption. The combiner using the decryption shares $\delta_{i_1}, \ldots, \delta_{i_t}$ from $\Gamma_{i_1}, \ldots, \Gamma_{i_t}$, respectively computes Δ by

$$\begin{aligned} \Delta &= \prod_{r=1}^{t} \delta_{i_r}^{\prod_{v=1, v\neq r}^{t} \frac{-i_v}{i_r - i_v}} \\ &= \prod_{r=1}^{t} (e(g, g)^{S x_2 \cdot f(i_r)})^{\prod_{v=1, v\neq r}^{t} \frac{-i_v}{i_r - i_v}} \\ &= e(g, g)^{\sum_{r=1}^{t} S x_2 \cdot f(i_r) \prod_{v=1, v\neq r}^{t} \frac{-i_v}{i_r - i_v}} \\ &= e(g, g)^{x_2 x_2^{-1} y S} \\ &= e(g, g)^{yS}. \end{aligned} \quad (3)$$

The message M can be computed by $M = B \cdot \Delta^{-1}$.

Verifiability.

1) Each user ID can verify their public key by computing $e(V^{ID}W, Z_{ID})$ and comparing it with $e(g, V)$.
2) Each server Γ_i verifies k_i by computing $e(V, g)^{k_i \cdot Y^{-w_i}}$ and comparing it to $v_i = e(V, g)^{f(i)}$.
3) Any server Γ_i can select t values $v_{j_1}, \ldots, v_{j_t}$ and compute

$$\begin{aligned} & \prod_{r=1}^{t} v_{i_r}^{\prod_{w=1, w\neq r}^{t} \frac{-i_w}{i_r - i_w}} \\ = & \prod_{r=1}^{t} (e(V, g)^{f(i_r)})^{\prod_{w=1, w\neq r}^{t} \frac{-i_w}{i_r - i_w}} \\ = & e(V, g)^{\sum_{r=1}^{t} f(i_r) \prod_{w=1, w\neq r}^{t} \frac{-i_w}{i_r - i_w}} \\ = & e(g, g)^{y} \\ = & e(g, Y). \end{aligned}$$

TTP Adds New Decryption Server. Assume that the TTP needs to add server Γ_{n+1}. The TTP computes $k_{n+1} = f(n+1) \cdot P_{n+1}^{y}$ and $v_{n+1} = e(V, g)^{f(n+1)}$.

TTP Removes a Decryption Server. Without loss of generality suppose the TTP needs to remove (or deactivate) server Γ_n, thus producing a t out of $n-1$ threshold decryption service [2]. First the TTP selects a new $x_{2,new} \in \mathbb{Z}_p^*$ and computes $V_{new} = g^{x_{2,new}}$ and $W_{new} = g^{x_1 x_{2,new}}$. Then for $i = 1, \ldots, n-1$ the TTP computes $k_{i,new} = f(i) \cdot P_i^{y}$ and $v_{i,new} = e(V, g)^{f(i)}$.

Server Γ_i Refreshes Their Secret Key. Suppose server Γ_i contacts the TTP and notifies them they wish to refresh the secret key. The server sends the TTP $P_{i,new}$. The TTP then selects a $R \in \mathbb{Z}_p^*$ and sends the challenge g^R to Γ_i. The server sends $g^{R \cdot DLOG(P_{i,new})}$ to the TTP. Here $DLOG(P_{i,new})$ is a u such that $g^u = P_{i,new}$. The TTP compares $e(g, g^{R \cdot DLOG(P_{i,new})})$ to $e(g^R, P_{i,new})$. If they are equal then the TTP updates k_i. Otherwise the sever Γ_i has lied and the TTP may punish (even remove the server).

TTP Removes a User with ID_0. Suppose that the TTP must dismiss user with identity ID_0. The TTP selects $x_{1,new} \in \mathbb{Z}_p^*$ and computes $W_{new} = g^{x_2 x_{1,new}}$ The TTP removes $g^{\frac{1}{ID_0 + x_1}}$ from the bulletin board. For all users ID with $ID \neq ID_0$ the TTP computes $g^{\frac{1}{ID + x_{1,new}}}$ and places it on the bulletin board.

4.2 Security Analysis

We assume the following security assumptions: Both the DLOG and CDH problems are hard in the presence in of a bilinear map. We assume that a threshold many servers act correctly. That is, if we have a t out of n threshold scheme then any t or many serves act correctly. If t or more servers are malicious then since they possess the threshold we assume that their actions are correct. Once a threshold is reached we cannot claim protection.

Theorem 1. *Given a coalition of less than t active servers, then the coalition cannot decrypt any validly constructed (using current public values) ciphertext C.*

Proof. Let $\rho < t$ and let $\Gamma_1, \Gamma, \ldots, \Gamma_\rho$ denote a set of ρ many active servers, a coalition which attempts to decrypt the server. Because shares have been distributed in a t out of n manner the coalition cannot decrypt the ciphertext without additional information beyond the shares distributed to the ρ servers. This additional information must come from servers who are no longer active (due to the threshold requirement). Let $\Phi_1, \ldots, \Phi_\omega$ denote deactivated servers who contribute (possibly actively or passively) with the coalition $\Gamma_1, \Gamma, \ldots, \Gamma_\rho$. Then $\omega + \rho \geq t$. Recall ciphertext $C = (W_{curr}^{S} \cdot V_{curr}^{S \cdot ID_0}, e(g, Y)^S \cdot M) = (A, B)$ where S is random, $W_{curr} = g^{x_1 x_{2,curr}}$ and $V_{curr} = g^{x_{2,curr}}$.

As $\Gamma_1, \Gamma, \ldots, \Gamma_\rho$ are active servers, they have shares $k_{\Gamma_j, curr}$ and $v_{\Gamma_j, curr}$ constructed for use with W_{curr} and V_{curr}. Now $\Phi_1, \ldots, \Phi_\omega$ are deactivated servers, they may possess "dated shares" (perhaps downloading from bulletin board earlier). They possess shares $k_{\Phi_i, time_i}$ and $v_{\Phi_i, time_i}$ constructed for use at time $time_i$ where W_{time_i} and V_{time_i}, note that $time_i < curr$. However the shares $k_{\Phi_i, time_i}$ and $v_{\Phi_i, time_i}$ do not work with shares $k_{\Gamma_j, curr}$ and $v_{\Gamma_j, curr}$ because $W_{curr} \neq W_{time_i}$ and $V_{curr} \neq V_{time_i}$ for all i. Therefore the only alternative is that at least t members from $\{\Gamma_1, \Gamma, \ldots, \Gamma_\rho, \Phi_1, \ldots, \Phi_\omega\}$ share some time $time_0$ such that each of these t members possess $k_{\Gamma_j, time_0}$ and $v_{\Gamma_j, time_0}$ or $k_{\Phi_i, time_0}$ and $v_{\Phi_i, time_0}$, respectively. Then for server i', this server i' will be able to compute $e(g, g)^{S x_{2,time_0} f_{time_0}(i')}$ where the constant coefficient of $f_{time_0}(x)$ is $x_{2,time_0}^{-1} y$. Now when all t servers apply their $\delta_{i'}$ into Equation (3), the corresponding Δ satisfies $\Delta = e(g, g)^{x_{2,curr} x_{2,time_0}^{-1} yS} \neq e(g, g)^{yS}$.

Therefore ρ many active servers, with $\rho < t$, cannot decrypt the ciphertext. □

Theorem 2. *If active server Γ_i refreshes their public key $P_{i,curr}$ then Γ_i knows the discrete log of $P_{i,curr}$.*

Proof. This follows directly from the refresh key protocol and the fact that the DLOG problem is "hard". □

4.3 Efficiency of Our Schemes

The the cost of executing the refresh properties are (in big O notation) is described in Table 2.

Here in Table 2, the value m represents the number of users and n represents the number of servers. In most applications one should expect that m is significantly larger

[2] We characterize the $n-1$ servers as active servers.

Table 2: Computational cost

operation	computational cost of our scheme	computational cost of the Long scheme
TTP refreshes secret key	$O(n)$	$O(mn)$
TTP adds new decryption server	$O(1)$	$O(m)$
TTP removes a decryption server	$O(n)$	$O(mn)$
server Γ_i refreshes their secret key	$O(1)$	$O(m)$
TTP removes a user with ID_0	$O(m)$	not discussed

than n. The cost is only for computational purposes there will also be a communication cost, although one may expect the communication cost between the TTP and the bulletin board is significantly less than the communication cost between the TTP and a server.

The above table demonstrates that our scheme is significantly more efficient than the existing schemes. Furthermore we have added a new service the dismissal of a user.

5 Conclusion

In this paper we have discussed dynamic threshold decryption scheme using the bilinear pairing. We have analyzed previous scheme noting their weaknesses, in particular their inefficiency. We have constructed a new scheme that is significantly more efficient than the previous schemes.

These schemes all rely on the use of a bulletin board to achieve the necessary dynamic properties. It remains an open problem if a dynamic public key scheme can be constructed without the use of a bulletin board. It remains an open problem if one can construct a dynamic scheme which uses bilinear pairing that allows the dismissal of users in $O(1)$ computations.

References

[1] J. Baek, J. Newmarch, R. S. Naini, and W. Susilo, "A survey of identity-based cryptography," in *Proceedings of Australian Unix Users Group Annual Conference*, pp. 95–102, 2004.

[2] G. R. Blakley, "Safeguarding cryptographic keys," in *Proceedings of AFIPS '79*, vol. 48, pp. 313–317, 1979.

[3] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology (Crypto'01)*, LNCS 2139, pp. 213–229, Springer-Verlag, 2001.

[4] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," *Journal of Cryptology*, vol. 17, no. 4, pp. 297–319, 2004.

[5] L. Chen, D. Gollmann, C. Mitchell, and P. Wild, "Secret sharing with reusable polynomials," in *Proceedings of the Second Australasian Conference on Information Security and Privacy (ACISP'97)*, pp. 183–193, Springer-Verlag, 1997.

[6] Y. Desmedt, "Society and group oriented cryptography: A new concept," in *Advances in Cryptology (Crypto'87)*, pp. 120–127, 1987.

[7] K. Kim, S. Lim, I. Yie, and K. Kim, "Cryptanalysis of a dynamic threshold decryption scheme," *Communications of the Korean Mathematical Society*, vol. 24, no. 1, pp. 153–159, 2009.

[8] C. S. Laih, L. Ham, J. Y. Lee, and T. Hwang, "Dynamic threshold scheme based on the definition of cross-product in an n-dimensional linear space," *Journal Information Science and Engineering*, vol. 7, pp. 13–23, 1991.

[9] Y. Long and K. F. Chen, "Construction of dynamic threshold decryption scheme from pairing," *International Journal of Network Security*, vol. 2, no. 2, pp. 111–113, 2006.

[10] A. Shamir, "How to share a secret," in *Communications of the ACM*, vol. 22, pp. 612–613, 1979.

[11] A. Shamir, "Identity based cryptosystems and signature schemes," in *Advances in Cryptology (Crypto'84)*, LNCS 196, pp. 47–53, Springer-Verlag, 1984.

[12] H. M. Sun and S. P. Shieh, "Construction of dynamic threshold schemes," *Electronics Letters*, vol. 30, pp. 2023–2025, 1994.

Brian King received a Ph.D. in mathematics (1990) and a Ph.D. in Computer Science (2000). He is currently an associate professor of Electrical and Computer Engineering at Indiana Univ. Purdue Univ. Indianapolis (IUPUI). Prior to joining IUPUI he worked in the Security Technologies Lab at Motorola Research Labs. His research interests include: wireless security, cryptography, threshold cryptography and low-complexity cryptosystems.