Threshold Password-Based Authentication Using Bilinear Pairings*

Songwon Lee[1], Kyusuk Han[1], Seok-kyu Kang[1], Kwangjo Kim[1] and So Ran Ine[2]

[1] Information and Communications University (ICU)
119, Munjiro, Yuseong-Gu, Daejeon, Korea
{swonlee, hankyusuk, redorb,kkj}@icu.ac.kr
[2] NITZ. Corp.
San 14-1 Koejong-dong, Seo-gu, Daejon, Korea
srine@nitz.co.kr

Abstract. We present a new threshold password-based authentication protocol that allows a roaming user(a user who accesses a network from different client terminals) to download a private key from remote servers with knowledge of only his identity and password. He does not need to carry the smart card storing his private information. We aim that a protocol has to allow a user to get his private key from the servers, even if some of the servers are compromised under the multi-server roaming system. In this paper, we firstly suggest a threshold password-only roaming protocol using (k,n)-*threshold scheme* which only k honest servers or more are engaged to reconstruct a secret key. Our scheme is based on bilinear pairings which could be built from Weil pairing or Tate pairing.

1 Introduction

With rapid development on the Internet a user Bob can easily access the network to get some services from a service provider, or to retrieve his sensitive private data stored in the server previously. In that case, he has to convince the server that he is a really legitimate user. To verify the identity(ID for short) of a user many real systems use password-based authentication. The fundamental problems with passwords come from the fact that most users' passwords are drawn from a relatively small spaces and are easily memorable, which also means that the password may be easily guessed by an attacker.

Let us assume that a roaming user accesses a network from different client terminals to download a private key from remote servers with knowledge of only his ID and password. He does not need to carry the smart card storing his private information. While the smart card plays an important role in storing sensitive information, it is impractical in many real environments due to inconvenience, for example, a user needs an external interface device to communicate with a smart card. Focused on this point, strong password authentication protocols were

* This work was partially supported by R&D Program for Fusion Strategy of Advanced Technologies, Ministry of Science and Technology in Korea

presented by Perlman *et al.*[16], Ford *et al.*[6], Jablon[9], *etc.* Some of them used multiple servers to implement a roaming protocol that uses a weak secret key to securely retrieve and reconstruct a strong private key that has been divided into pieces distributed among multiple servers. We note that as one of goals, a protocol has to allow a user to get his private key from the servers, even if some of the servers are compromised.

In this paper we present a threshold password-based authentication protocol for a roaming user. We use a (k,n)-*threshold scheme* in which only k honest servers or more are engaged to reconstruct a secret key. Our scheme is based on bilinear pairings that could be built from Weil pairing or Tate pairing over *Gap Diffie-Hellman(GDH)* group, which *Computational Diffie-Hellman(CDH)* problem is hard but *Decision Diffie-Hellman(DDH)* problem is easy.

2 Preliminaries

2.1 Previous Works

Perlman and Kaufman [16] presented protocols that we can securely retrieve a private key and use this to download the rest of our security context. Ford and Kaliski [6] proposed methods that use multiple servers to prevent attacking by introducing *password hardening protocol* by which servers interact with the user to harden the user's password into a strong secret without revealing either the user's password or the hardened result. A *password-only multi-server roaming protocol* is presented by Jablon [9]. In his protocol, the user can authenticate servers and retrieve his private key for temporary use on a client machine using just an easily memorable password. [6] and [9] make use of the multiple servers to gain the goal of the protocol. When some of the servers are compromised, the user can not obtain valid secret key no matter what the user has a method to verify the key. In our proposed scheme, we deal with this problem.

Let's briefly review here the protocol proposed in [9]. In this protocol the author introduced *forgiveness protocol* by which user's honest mistakes are forgiven by sending evidence of recent prior invalid access attempts after each successful authentication. But we do not consider here this forgiveness.

Parameters. The protocol operates in a subgroup of order q in $\mathbb{Z}_p^*$,where p, q and r are odd primes, $p = 2rq + 1, 2^{k-1} < p < 2^k, r \neq q$, and $2^{2_j - 1} < q < 2^{2_j}$.
Enrollment. The user, Alice, selects a password π, computes $g_\pi = h(\pi)^{2r}$, and creates a private key U. For each $i \in [1, n]$, she computes a secret key share $S_i = {g_\pi}^{y_i}$ using randomly chosen $y_i \in_{\mathcal{R}} [1, q-1]$. She then generates her master j-bit symmetric key with $K_m = h(S_1 \parallel \cdots \parallel S_n) \bmod 2^j$, her encrypted private key as $U_K =_{K_m} \{U\}$, and her key verifier $proof_{PK_m} = h(K_m \parallel g)$.

1. *Client*: send $\{ID_A, y_i, U_K, proof_{PK_m}\}$ to each server L_i for all $i \in [1, n]$.
2. *Servers*: store them in a list C_i maintained on each server.

Authenticated Retrieval. To retrieve her master key at a later time, the client and servers perform the protocol as below:

1. *Client*: select a random number $x \in [1, q-1]$, computes $X = {g_\pi}^x \bmod p$, and then send $\{ID_A, X\}$ to *Servers*.
2. *Servers*: retrieve $\{ID_A, y_i, U_K, proof_{PK_m}\}$ from C_i, compute $Y_i = X^{y_i}$, and then reply $\{Y_i, U_K, proof_{PK_m}\}$ to *Client*.
3. *Client*: compute $S_i = Y_i^{1/x} \bmod p$ for each $i \in [1, n]$, and then generate $K' = h(S_1 \parallel S_2 \parallel \cdots \parallel S_n)$. If $proof_{PK_m} \neq h(K' \parallel g)$ abort, otherwise compute $U =_{1/K'} \{U_K\}$.

2.2 Threshold Cryptosystem

The concept of a threshold scheme, called secret sharing scheme was introduced in [17] and since then many researchers have investigated such schemes.

A (k,n)-*threshold secret sharing scheme* is a protocol among n players in which the *dealer* distributes partial information (*share*) about a *secret* to n participant such that: a) Any group of fewer k participants can not obtain any information about the secret. b) Any group of at least k participants can compute the secret in polynomial time.

We use the verifiable secret sharing (VSS) scheme by Pedersen [20], denoted as *Pedersen-VSS*. The next lemma summarizes some properties of *Pedersen-VSS*, which are used in the analysis of our protocol.

Lemma 1. ([20]) Pedersen-VSS *satisfies the following properties in the presence of an adversary that corrupts at most k-1 parties and which cannot compute* $\log_g h$*:*

1. *If the dealer is not disqualified during the protocol then all honest players hold shares that interpolate to a unique polynomial of degree k-1. In particular, any k of these shares suffice to efficiently reconstruct (via interpolation) the secret s.*
2. *The protocol produces information (the public values* E_i *and private value* s_i*) that can be used at reconstruction time to test for the correctness of each share; thus, reconstruction is possible, even in the presence of malicious players, from any subset of shares containing at least k correct shares.*
3. *The view of the adversary is independent of the value of the secret s, and therefore the secrecy of s is unconditional.*

In the next section, we describe our proposed password-based authentication scheme making use of the (k,n)-*threshold scheme* in which a user distributes secrets to multiple servers, assuming $n \geq 2k - 1$ [20, 10].

2.3 Bilinear Pairings

Let us consider an additive group $\mathbb{G}_1$ and a multiplicative group $\mathbb{G}_2$ of the same order q. Assume that the discrete logarithm problem is hard in both groups. Let P be a generator of $\mathbb{G}_1$, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ a bilinear map satisfying the following properties:

1. *Bilinearity*: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and all $a, b \in \mathbb{Z}$.
2. *Non-degeneracy*: if $\hat{e}(P, Q) = 1$ for all $Q \in \mathbb{G}_1$, then $P = \mathcal{O}$.
3. *Computability*: there exists an efficient algorithm to compute $\hat{e}(P, Q)$ for any $P, Q \in \mathbb{G}_1$.

With such groups $\mathbb{G}_1$ and $\mathbb{G}_2$, we can define hard cryptographic problems like *Computational Diffie-Hellman (CDH)* problem, *Decision Diffie-Hellman (DDH)* problem, and *Gap Diffie-Hellman(GDH)* problem. We skip detailed definition of these problems. To construct the bilinear pairing, we can use the Weil pairing and Tate pairing. $\mathbb{G}_1$ is a cyclic subgroup of the additive group of points of a supersingular elliptic curve $E(\mathbb{F}_p)$ over a finite field while $\mathbb{G}_2$ is a cyclic subgroup of the multiplicative group associated to a finite extension of $\mathbb{F}_p$.

3 Our Proposed Scheme

3.1 Model

Our model for multi-server roaming protocol is similar to that of [9], but with some different features. First, our scheme employs the concept of threshold scheme, where the user plays the role of a dealer to distribute secret shares to n servers. To do this, we make use of the Pedersen-VSS protocol [20] in a different way that only the user who knows an extra information, *password*, can obtain the secret value derived from the password in collaboration with threshold servers. While the protocol in [9] uses a n-out-of-n solution, *i.e.*, the password information is shared among n servers and they *all* must cooperate to authenticate the user, the protocol in our model uses k-out-of-n solution. In addition, even if an adversary compromises k or more servers, she cannot reconstruct the secret value, without knowing user's password. Second, our scheme is based on bilinear pairings that can be built from Weil pairing or Tate pairing over *GDH* group, which *CDH* problem is hard but *DDH* problem is easy. On the other hand, although we don't consider *forgiveness protocol* introduced in [9] by which user's honest mistakes are forgiven by sending evidence of recent prior invalid access attempts after each successful authentication, this forgiveness can be adapted in our system. Figure 1 depicts the concept of our model.
There are two phases: (1) Enrollment phase — The user, U enrolls his credentials in the servers at his own client terminal, C which may hold user's private information. (2) Retrieval phase — The user may move to other place where he is just able to use different client terminal, C' which does not hold any user-specific information. The protocol, however, allows the user to download a private key from remote servers with knowledge of only his identity and password.

Enrollment [$\mathcal{TPS}$ Protocol]. After constructing (k,n)-threshold system as similar as in Pedersen-VSS, the client terminal (For simplicity, we say "client ".) creates n shares using the Pedersen-VSS scheme. k shares will contribute to reconstruct the master symmetric key K_m which is derived from a user's password π. Then,

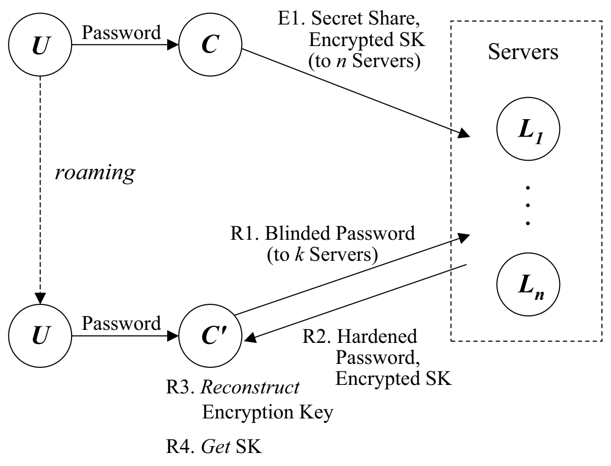
U
Password
C
E1. Secret Share, Encrypted SK (to n Servers)
Servers
L1
roaming
R1. Blinded Password (to k Servers)
Ln
U
Password
C'
R2. Hardened Password, Encrypted SK
R3. Reconstruct Encryption Key
R4. Get SK

Fig. 1. The concept of our model

the client have user's private key SK encrypted with the symmetric key K_m. Finally, he creates a proof value V that links the password to his master key.

The client sends secretly share Y_i, the encrypted private key U_K, and the proof value V to each server.

As in [9], the enrollment protocol flow has to be performed through a secure channel that authenticates the identity of the user to each i^{th} server L_i.

Authenticated Retrieval [$\mathcal{TPR}$ Protocol]. When at any available client terminal, the user wants to download his private key stored in the server, the client first performs the threshold protocol with at least k servers. Note here that no client terminal has a user's information created at enrollment time.

The client randomly chooses at least k servers and sends them a randomly blinded form of the password to each server. On receiving the request, each server in turn responds with a blinded reply consisting of the blinded password. At least one of the servers also sends the encrypted private key U_K and its proof value V to the client.

The client reconstructs user's master key K_m using the shares and user's password, and then verifies whether the master key is correct using the proof value V and the master key K_m. Finally, if the master key is correct, the user gets his private key SK by decrypting U_K with the master key.

3.2 Definitions

Communication Model. We assume that our computation model is composed of a set of n players $\{L_1, L_2, \ldots, L_n\}$. They are connected by a complete network of secure point-to-point channels. The players have access to a dedicated broadcast channel in which when a player sends a broadcast message all other players can receive the message and know exactly from whom the message sent.

The Adversary. We assume that there exists an adversary, $\mathcal{A}$, who can corrupt up to $k-1$ of n servers in the network, where $n \geq (2k-1)$. We consider a malicious adversary that may cause corrupted players to divert from the specified protocol in any way. We assume that the computational power of the adversary is adequately modeled by a probabilistic polynomial time Turing machine. Our adversary is *static*, *i.e.*, chooses the corrupted players at the beginning of the protocol.

Now, we state some definitions similar in [11, 12], for the analysis of our protocol.

In the following we assume that there are a dealer C and n players $\{L_1, L_2, \ldots, L_n\}$. We say that C `broadcasts` a message m, when she puts m on the broadcast channel for everybody to hear it. In particular $\mathcal{A}$ can hear the message too. We say that C `distributes` a message if she puts m on the private channels connecting her to all the other players. Notice that $\mathcal{A}$ can see m only if somebody has been corrupted.

Let $\mathcal{P}$ be a pair of protocols where the second is always executed after the first one, $\mathcal{P}$=`(Share-Verify, Recover)` for the players $\{L_1, L_2, \ldots, L_n\}$ and a dealer C.

Definition 1. (View) *The* adversary view, $\text{View}^{\mathcal{P}}_{Network,\mathcal{A}}(\cdot)$ *during protocol* $\mathcal{P}$ *is the probability distribution over the set of computational histories (traffic and coin tosses) of the bad players.*

Sometimes we accompany some distributed protocol $\mathcal{P}$ we propose by a description of a *simulator* `Sim` which is needed to analyze the security of this protocol. `Sim` is an algorithm that plays the role of the honest players. $\mathcal{A}$ interacts with `Sim` as if she was interacting with the network. `Sim` tries to create a view for $\mathcal{A}$ that is indistinguishable form the real one. That is, the process of simulation is a computation of two interactive algorithms, `Sim` and $\mathcal{A}$, where the simulator controls the uncorrupted players, and $\mathcal{A}$ controls the corrupted players. Therefore a description of a simulation process is similar to a description of the protocol itself [13].

Definition 2. *The protocol* $\mathcal{P}$ *is called k-secure (or secure with threshold k) if in the presence of an attacker that corrupts at most k-1 parties the following requirements for correctness and secrecy are satisfied:*

Correctness:
1. *All subsets of k shares provided by honest players define the same unique secret value.*
2. *Secret value is uniformly distributed.*

Secrecy: (Simulatability) *For every (probabilistic polynomial-time) adversary* $\mathcal{A}$*, there exists a (probabilistic polynomial-time) simulator* `Sim`*, such that on input an element Y, produces an output distribution which is polynomially* *indistinguishable from $\mathcal{A}$'s view of a run of $\mathcal{P}$ that ends with Y as its output, and where $\mathcal{A}$ corrupts up to k-1 parties. That is, the adversary view* $\mathrm{View}^{\mathcal{P}}_{Network,\mathcal{A}}(\cdot)$ *is identical with* $\mathrm{View}^{\mathcal{P}}_{Sim,\mathcal{A}}(\cdot)$ *which is the adversary view of the simulated execution of the protocol.*

A simulator of each subprotocol exhibits the secrecy of this subprotocol, which states that the adversary learns nothing from the protocol beyond the public inputs and outputs of this protocol, or in other words, that the adversary learns as much by participating in the threshold computation as he would learn from observing this operation as a block-box.

We now come up with the following definition of *the secure threshold password-authenticated key retrieval protocol* (TPKR for short).

Definition 3. *In our* TPKR $= (\mathcal{TPS}, \mathcal{TPR})$*, Let two kinds of* (`static`) *adversaries exist as follows:*

1. *Strong Adversary: the adversary is able to corrupt and to control k or more servers if he desires.*
2. *Weak Adversary: Not strong adversary, i.e., the adversary is just able to corrupt at most k-1 servers.*

Definition 4. (**Secure** TPKR) *Let TPKR be the (k,n)-threshold password-authenticated key retrieval protocol, where 2k-1≤n. TPKR is a strongly secure protocol if:*

1. *The protocol is k-secure satisfying Definition 2 in the presence of the weak adversary in Definition 3.*
2. *No adversary, even strong one, without knowing user's password is able to reconstruct the master symmetric key K_m, and is thus able to obtain the private key SK.*

3.3 Detailed Protocol

We let ℓ be the security parameter given to the setup algorithm and let $\mathcal{G}$ be some GDH parameter generator.

System Setup Given a security parameter ℓ, the algorithm $\mathcal{G}$ works as follows:

1. Run $\mathcal{G}$ on input ℓ to generate a prime $q \geq 2^{\ell}$, two cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of the same order q and a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.
2. Choose two arbitrary generators P and $P' \in \mathbb{G}_1$, where $P' = \alpha P$ for some $\alpha \in \mathbb{Z}_q$ and the computing α given P and P' is infeasible.
3. Choose cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{G}_1$, $H_2 : \mathbb{G}_2 \to \{0,1\}^{\kappa}$, and $H_3 : \{0,1\}^{\kappa} \times \mathbb{G}_1 \to \mathbb{Z}_q^*$, for some κ. The security analysis will view H_1, H_2, and H_3 as random oracles.
4. The system parameters are $\mathsf{params} = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, H_1, H_2, H_3, P, P'\}$. And then publish them.

Enrollment Protocol $\mathcal{TPS}$ The enrollment protocol makes use of Pedersen-VSS protocol, but we introduce elliptic curve cryptosystems for bilinear parings.

Denote n servers involving in the protocol as $\{L_1, L_2, \ldots, L_n\}$ and the client playing the role of a dealer as C. Let ID and SK be the user's identity and the private key, respectively. The user having ID picks his password π. We assume that π can be mapped into $\mathbb{Z}_q$. The user then performs the following protocol at the client terminal as follows:

1. The client C, as a dealer, distributes user's credentials.
   (a) Select randomly y and $z \in \mathbb{Z}_q^*$.
   (b) Choose two random polynomials $f(x)$ and $g(x)$ over $\mathbb{Z}_q$ of degree $k-1$ such that $f(0) = a_0 = y$ and $g(0) = b_0 = z$. Let
   $$f(x) = a_0 + a_1 x + \cdots + a_{k-1}x^{k-1} \quad \text{and}$$
   $$g(x) = b_0 + b_1 x + \cdots + b_{k-1}x^{k-1}.$$
   (c) Compute and broadcast $E_i = E(a_i, b_i) = a_i P + b_i P'$ for $i = 0, \ldots, k-1$.
   (d) Compute $K = \hat{e}(yR_{ID}, Q_{ID})$, then create the master symmetric key $K_m = H_2(K)$, where $R_{ID} = H_1(\pi)$ and $Q_{ID} = H_1(ID)$. Then create the encrypted private key $U_K = E_{K_m}(SK)$ and the key verifier $V = H_3(K_m, P)$, and let $Y_i = f(i) \cdot Q_{ID}$ and $Z_i = g(i) \cdot Q_{ID}$ for $i = 1, 2, \ldots, n$.
   (e) Send $\{ID, Y_i, Z_i, U_K, V\}$ *secretly* to each player L_i for all $i \in [1, n]$.
2. After receiving all the information from C, L_i does as follows.
   (a) Verifies that
   $$\hat{e}(Y_i, P) \cdot \hat{e}(Z_i, P') \stackrel{?}{=} \hat{e}\left(Q_{ID}, \sum_{j=0}^{k-1} i^j \cdot E_j\right). \qquad (1)$$
   (b) If Eq.(1) is verified to be false, response a *complaint* against C. Otherwise accept and store $\{ID, Y_i, U_K, V\}$ in a storage maintained on each L_i.
3. C discards all information, and completes the enrollment protocol.

For the sake of convenience, we assume that the client has received no complaint in Step 2.

Authenticated Retrieval Protocol $\mathcal{TPR}$ For authenticated retrieval, the client and k servers perform the actions as follows: Denote k servers by $B = \{L_i \mid 1 \leq i \leq k\}$.

1. C sends k servers a request message.
   (a) Select a random number uniformly distributed $x \in \mathbb{Z}_q^*$, compute $X = xR_{ID}$.
   (b) Send X and ID to each server L_i for $i \in B$.
2. On receiving the request, each server L_i responds as follows:

   (a) Retrieve $\{ID, Y_i, U_K, V\}$ from the storage maintained securely on each L_i.
   (b) Compute $R_i = \hat{e}(X, Y_i)$, and then reply $\{R_i, U_K, V\}$ to the client.
3. Finally, the client reconstructs user's private key by performing the following:
   (a) Compute $l_i = \prod_{j \in B, j \neq i} \frac{j}{j-i}$ for each i^{th} server.
   (b) Compute $R'_i = (R_i)^{l_i x^{-1}}$ and $K' = \prod_{i \in B} R'_i$.
   (c) Generate $K'_m = H_2(K')$.
   (d) If $V \neq H_3(K'_m, P)$, abort.
   (e) To obtain the private key, decrypt U_K with the master key K'_m.

Completing the protocol successfully, the client reconstructs the user's private key SK. Figure 2 depicts the retrieval protocol.

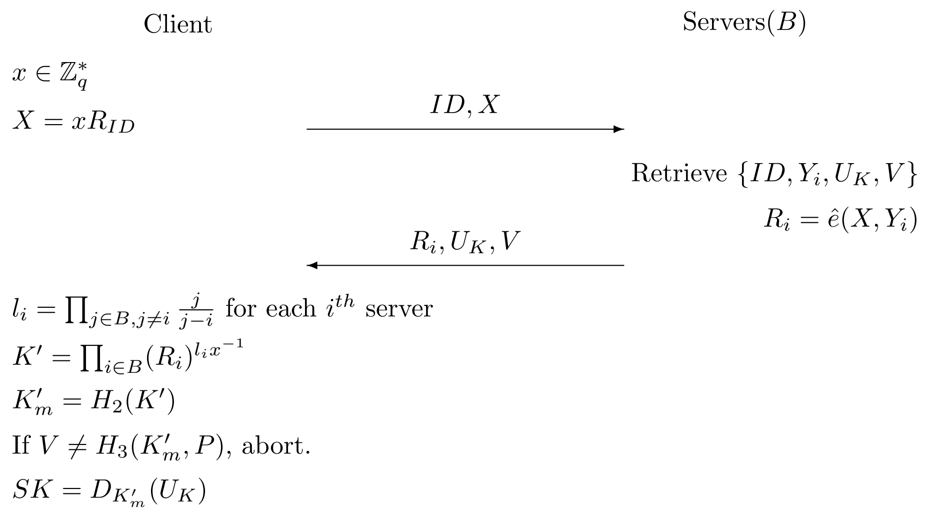
Client
Servers(B)
$x \in \mathbb{Z}_q^*$
$X = xR_{ID}$
ID, X
Retrieve $\{ID, Y_i, U_K, V\}$
$R_i = \hat{e}(X, Y_i)$
R_i, U_K, V
$l_i = \prod_{j \in B, j \neq i} \frac{j}{j-i}$ for each i^{th} server
$K' = \prod_{i \in B} (R_i)^{l_i x^{-1}}$
$K'_m = H_2(K')$
If $V \neq H_3(K'_m, P)$, abort.
$SK = D_{K'_m}(U_K)$

Fig. 2. Our Threshold Key Retrieval Protocol

4 Security Analysis

In this section, we discuss with the security aspects of our proposed scheme TPKR and give the complete security proof. As a result, the security for our protocol arrives at the security for secret value K, assuming that the adapted symmetric cryptosystem is secure and thus nobody can obtain the private key SK without knowing K. Note that $UK = E_{K_m}(SK)$, where $K_m = H_2(K)$.

Lemma 2. (Correctness) *The protocol* TPKR=($\mathcal{TPS}$, $\mathcal{TPR}$) *from Section 3.3 satisfies the correctness of Definition 2 with threshold k, for any* $k \leq (n+1)/2$.

Proof. First note that all the honest players indeed hold the verification equation Eq.(1) as follows:

$$\hat{e}(Y_i, P) \cdot \hat{e}(Z_i, P') = \hat{e}\Big(Q_{ID}, \sum_{j=0}^{k-1} i^j \cdot E_j\Big).$$

Since

$$\hat{e}(Y_i, P) \cdot \hat{e}(Z_i, P') = \hat{e}(f(i)Q_{ID}, P) \cdot \hat{e}(g(i)Q_{ID}, P') = \hat{e}(Q_{ID}, f(i)P + g(i)P'),$$

where

$$f(i)P + g(i)P' = \sum_{j=0}^{k-1} i^j (a_j P + b_j P') = \sum_{j=0}^{k-1} i^j E_j.$$

1. From **Lemma 1.1**, we know that all honest players hold shares (Y_i) which contribute to reconstruct unique secret by combining with client's request message as in step 2 of $\mathcal{TPR}$ protocol. For any set $\mathcal{B}$ of k shares and extra value (X) from client's request message, the unique secret is computed as follows:

$$\begin{aligned} K' &= \prod_{i \in \mathcal{B}} \hat{e}(X, Y_i)^{l_i x^{-1}} = \prod_{i \in \mathcal{B}} \hat{e}(xR_{ID}, f(i)Q_{ID})^{l_i x^{-1}} = \prod_{i \in \mathcal{B}} \hat{e}(f(i) l_i R_{ID}, Q_{ID}) \\ &= \prod_{i \in \mathcal{B}} \hat{e}(f(i) \prod_{j \in \mathcal{B}, j \neq i} \frac{j}{j-i} R_{ID}, Q_{ID}) = \hat{e}(\sum_{i \in \mathcal{B}} f(i) \prod_{j \in \mathcal{B}, j \neq i} \frac{j}{j-i} R_{ID}, Q_{ID}) \\ &= \hat{e}(yR_{ID}, Q_{ID}), \end{aligned}$$

where l_i is an appropriate Lagrange interpolation coefficient for the set $\mathcal{B}$. Since the above holds for any set of k correct shares then K' is uniquely defined, where the same extra value (X) which as said is derived from the user's password has to be given to the protocol $\mathcal{TPS}$ and $\mathcal{TPR}$.

2. Let's consider the secret value K. We can let K be $g^{\lambda y}$ for some λ where g is a generator of group $\mathbb{G}_2$. Since y is chosen randomly from $\mathbb{Z}_q^*$ as in [20], therefore $K = g^{\lambda y}$ is also a random element in the group $\mathbb{G}_2$. On the other hand, from **Lemma 1.3**, the view and thus actions of the adversary are independent of the secret y.

As a result, we can state that TPKR can be resistant to corruption of even $k-1$ of $n \geq 2k-1$ servers. A user chooses randomly a secret value y uniformly distributed in $\mathbb{Z}_q^*$ during the execution of $\mathcal{TPS}$. Even there exists an adversary who can corrupt at most $k-1$ servers among $n \geq 2k-1$, any subset of k servers constructs the secret value K uniformly distributed in $\mathbb{G}_2$.

Lemma 3. (Secrecy) *Protocol* TPKR *from Section 3.3 satisfies the secrecy of Definition 2.*

Proof. We can prove in a similar way which is used to prove Lemma 3 in [13]. We can state that, for any input secret y and y', if the dealer $\mathcal{C}$ is uncorrupted then there is no difference between the adversarial view of an execution of TPKR in which $\mathcal{C}$ shares y', from a view of TPKR in which $\mathcal{C}$ shares y.

There exists Sim such that for every $n/2$-threshold static secure-channels adversary $\mathcal{A}$ with history $h_{\mathcal{A}}$, for any given system parameter $\texttt{params}$. Let E stand for an instance (P, P') of Pedersen commitment [20]. Let $f(x), g(x)$ be any $k-1$ degree polynomials such that $f(0) = y$. Consider a run of TPKR in which $\mathcal{C}$ uses polynomials $f(x), g(x)$ and the random input of $\mathcal{A}$ is $r_{\mathcal{A}}$. Note that once we fix $f(x), g(x)$, $r_{\mathcal{A}}$ then everything else in the run of this protocol is determined. Denote the outputs of such run as $\mathcal{TPKR}_{\mathcal{A}}^{Data}$ $((h_{\mathcal{A}},\ r_{\mathcal{A}}),\ E; f(x), g(x))$. We denote the set of corrupted players as $P_{\mathcal{B}}$ and the set of uncorrupted players as $P_{\mathcal{G}}$.

Note that any $k-1$ degree polynomials $f'(x), g'(x)$ such that $f'(i) = f(i)$ for $L_i \in P_{\mathcal{B}}$ and $f(x) + \alpha g(x) = f'(x) + \alpha g'(x)$ where $P' = \alpha P$, the adversary's output in $\mathcal{TPKR}_{\mathcal{A}}^{Data}$ $((h_{\mathcal{A}}, r_{\mathcal{A}}),\ E; f'(x), g'(x))$ is the same as in $\mathcal{TPKR}_{\mathcal{A}}^{Data}$ $((h_{\mathcal{A}}, r_{\mathcal{A}}),\ E; f(x), g(x))$.

If we fix $y, y', r_{\mathcal{A}}$, and range the polynomials $f(x), g(x)$ among all $k-1$ degree polynomials such that $f(0) = y$, then we see that the distribution of the adversary view in the following two cases are equal, for every y, y', and $r_{\mathcal{A}}$:

1. TPKR on $f'(x), g'(x)$, and $r_{\mathcal{A}}$ followed by protocol on the resulting $\mathcal{TPKR}_{\mathcal{A}}^{Data}$, where $f'(x), g'(x)$ are random $k-1$ degree polynomials such that (0) $f'(0) = y'$; (1) $f'(i) = f(i)$ on $L_i \in P_{\mathcal{B}}$; (2) $f'(x) + \alpha g'(x) = f(x) + \alpha g(x)$ where $f(x), g(x)$ are random $k-1$ degree polynomials such that $f(0) = y$.
2. TPKR on $f(x), g(x)$, and $r_{\mathcal{A}}$ with outputs denoted $\mathcal{TPKR}_{\mathcal{A}}^{Data}[y]$, followed by protocol on inputs $\mathcal{TPKR}_{\mathcal{A}}^{Data}[y']$ output by replacement procedure $\mathcal{T}_{TPKR}$ $(\mathcal{TPKR}_{\mathcal{A}}^{Data}[y], y', \alpha)$ where $f(x), g(x)$ are random $k-1$ degree polynomials such that $f(0) = y$.
   $\mathcal{T}_{TPKR}(\mathcal{TPKR}_{\mathcal{A}}^{Data}[y], y', \alpha)$ mentioned above takes as inputs a given secret-sharing $\mathcal{TPKR}_{\mathcal{A}}^{Data}[y]$, target value y' and α s.t. $P' = \alpha P$, and outputs its replacement $\mathcal{TPKR}_{\mathcal{A}}^{Data}[y']$ as in [13]. Note that the data which is visible to $\mathcal{A}$, *i.e.*, the public data and the private data of the players controlled by the $\mathcal{A}$, must remain the same in $\mathcal{TPKR}_{\mathcal{A}}^{Data}[y]$ and $\mathcal{TPKR}_{\mathcal{A}}^{Data}[y']$, so this *replacement* is always only a modification of the private data of the players controlled by the simulator.

From the above observations, (1) since $f(x)$ is a random polynomial such that $f(0) = y$, then $f'(x)$ is a random polynomial such that $f(0) = y'$; and (2) there is a one-to-one mapping between a choice of $g(x)$ and a choice of $g'(x)$, and thus since $g(x)$ is a random polynomial then so $g'(x)$.

From Lemmas 2 and 3, we can state the following theorem:

Theorem 1. *Assume that $n \geq 2k-1$. Then the protocol* TPKR *is k-secure threshold-authenticated roaming protocol according to Definition 2 with fault-tolerance k.*

The following lemma shows the security of the protocol $\mathcal{TPR}$ against a strong adversary who corrupts k or more servers if he desires.

Lemma 4. *Under GDH assumption, the protocol $\mathcal{TPR}$ is secure against a strong adversary defined by Definition 3.*

Proof. Let a *strong adversary* $\mathcal{A}$ be able to corrupt k or more servers and thus to obtain at least k shares. In this case, we need to show that $\mathcal{A}$ can not reconstruct the secret value K' without knowing the user's password.

In order to break the protocol, $\mathcal{A}$ tries to compute K'' such that

$$K'' = \prod_{i \in S} \hat{e}(R', Y_i)^{l_i}, \tag{2}$$

with knowledge of system parameter params, any set $\mathcal{S}$ of t secret shares Y_i for $i = 1, 2, \dots, t$ ($t \geq k$) and client's request messages X, where l_i is an appropriate Lagrange interpolation coefficient for the set $\mathcal{S}$.

We assume $\mathcal{A}$ has three options to compute K'': (1) Solve DLP, *i.e.* find an integer n such that $Q = nP$, (2) Solve CDHP in such a way that given $(P, \alpha P, \beta P) \in \mathbb{G}_1$ compute $\alpha\beta P$, and (3) Guess correct password, *e.g.*, by mounting password guessing attack.

1. In order to compute Eq.(2), $\mathcal{A}$ first unblinds X to obtain R', *i.e.* find an integer x (or $x' = x\beta$) such that $R' = xR_{ID}$ (or $R' = x'P$). Thus no adversary can compute R', under the assumption DLP is hard.

2. Let $Q_{ID} = \alpha P$, $R_{ID} = \beta P$. Even given a triple $(P, \alpha P, y\beta P)$, $\mathcal{A}$ can not compute $\alpha\beta y P$, such that $K'' = \hat{e}(yR_{ID}, Q_{ID}) = \hat{e}(P, P)^{\alpha\beta y} = g^{\alpha\beta y}$ where g is a generator of $\mathbb{G}_2$, assuming that $\mathbb{G}_1$ is a GDH group.

On the other hand, given $\{Y_i = y_i Q_{ID} \mid i = 1, 2, \dots, t\}$, $\mathcal{A}$ can compute the following at the best:

$$Y'' = \prod_{i \in S} \hat{e}(y_i Q_{ID}, P)^{l_i} = \hat{e}(yQ_{ID}, P) = \hat{e}(P, P)^{\alpha y} = g^{\alpha y}.$$

3. Let π' be a password guess by $\mathcal{A}$. Thus $R' = H_1(\pi')$. Now, $\mathcal{A}$ computes the following:

$$K'' = \prod_{i \in S} \hat{e}(R', Y_i)^{l_i} = \prod_{i \in S} \hat{e}(R', f(i)Q_{ID})^{l_i} = \hat{e}(yR', Q_{ID}),$$

However, $\mathcal{A}$ can not verify whether his guess is correct or not. More over, by Lemma 3 it is impossible to distinguish K'' from K'.

Theorem 2. *Assume $n \geq 2k - 1$. Then the protocol* TPKR *is strongly secure according to Definition 4, under GDH assumption.*

Proof. The proof of the theorem comes immediately from Theorem 1 and Lemma 4.

5 Comparison

With mainly compared to [6] and [9], our scheme is capable of resisting that fewer than threshold servers are compromised. When only k honest servers are involved in the protocol, the user can retrieve his private key with knowledge of his own password. Besides, even attacker has succeeded in compromising k or more servers but without knowing the user's password, she still cannot obtain any information about the user's credential.

Table 1 depicts computation load of TPKR compared with [9]. We denote $\mathbf{E}$ and $\mathbf{M}$ by computation load for exponentiation and multiplication, respectively. Let n be the number of servers and k be threshold value.

Table 1. Computation in the Retrieval on the Client Side

	Jab01 [9]	TPKR
Main computation parts	$S_i = R_i^{x^{-1}}$ $K' = h(S_1 \parallel \cdots \parallel S_n)$	$R'_i = R_i^{l_i x^{-1}}$ $K' = h(\prod_{i \in B} R'_i)$
Computation load	$n\mathbf{E}$	$k(\mathbf{E}+\mathbf{M})$

From Table 1, we see that our scheme TPKR is more efficient one than *Jab01* with respect to the computation during the retrieval, since the inequality $n\mathbf{E} \geq k\mathbf{E}+k\mathbf{M}$ holds, where $n \geq 2k-1$, *i.e.*, $(k-1)\mathbf{E} \geq k\mathbf{M}$ if $k \geq 2$. However with respect to the server side, the computation load of our protocol may be less efficient due to pairing operation which costs several times expensive than the exponentiation [21].

6 Conclusions

Based on pairings, we firstly suggest a new threshold password-only roaming protocol which allows a roaming user to download a private key from remote servers, without revealing the password to off-line guessing. No client terminal has his information created at enrollment time.

We note that, as a goal of a multi-server roaming system, a protocol has to allow a user to get his private key from the servers, even if some of the servers are compromised. In this paper, we use (k,n)-*threshold scheme* in which only k honest servers or more are engaged to reconstruct a secret key based on bilinear pairings that could be built from Weil pairing or Tate pairing. The protocol is useful for authenticating roaming users and even non-roaming users, and retrieving private keys for use in other applications. A design of more efficient schemes based on parings is left as one of further works.

Acknowledgement

The authors are very grateful to the anonymous reviewers for their valuable suggestions and comments to this paper.

References

1. S.Al-Riyami and K.Paterson, "Certificateless Public Key Cryptography", available at $http : \backslash\backslash www.ime.usp.br/ \sim rt/cranalysis/CertifLessPKC.pdf$, Jul.2003.
2. D.Boneh and M.Franklin, "Identity-Based Encryption from the Weil Pairing", *CRYPTO2001*, LNCS 2139, pp.213-229, Springer-Verlag, 2001.
3. S.Bellovin and M.Merritt, "Encrypted Key Exchange: Password-based Protocols Secure against Dictionary Attacks", *Proc. of IEEE Symposium on Research in Security and Privacy*, May 1992.
4. S.Bellovin and M.Merritt, "Augmented Encrypted Key Exchange: A Password-based Protocol Secure Against Dictionary Attacks and Password File Compromise", Technical Report, AT&T Bell Laboratories, 1994.
5. J.Baek and Y.Zheng, "Identity-Based Threshold Decryption", *IACR eprint*, 2003/164.
6. W.Ford and B.Kaliski, "Server-Assisted Generation of a Strong Secret from a Password", *Proc.*9^{th} *International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise*, IEEE, Jun.14-16, 2000.
7. F.Hess, "Efficient Identity Based Signature Schemes Based on Pairings", *SAC2002*, LNCS 2595, pp.310-324, Springer-Verlag, 2003.
8. D.Jablon, "Strong Password-Only Authenticated Key Exchange", *ACM Computer Communications Review*, Oct.1996.
9. D.Jablon, "Password Authentication Using Multiple Servers", *CT-RSA2001*, LNCS 2020, pp.344-360, Springer-Verlag, 2001.
10. B.Libert and J.Quisquater, "Efficient Revocation and Threshold Pairing Based Cryptosystems", *PODC'03*, pp.163-171, Jul.13-16, 2003.
11. R. Gennaro, "Theory and Practice of Verifiable Secret Sharing", PhD Thesis, MIT, May 1996.
12. R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin, "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems", *Advances in Cryptology - EUROCRYPT '99*, LNCS 1592, pp.295-310, Springer-Verlag, 1999.
13. S. Jarecki, "Efficient Threshold Cryptosystems", PhD Thesis, MIT, June 2001.
14. P.MacKenzie, T.Shirmpton and M.Jakobsson, "Threshold Password-Authenticated Key Exchange(Extended Abstract)", *CRYPTO2002*, LNCS 2442, pp.385-400, Springer-Verlag, 2002.
15. T.Pedersen, "Non-interactive and Information Theoretic Secure Verifiable Secret Sharing", *CRYPTO'91*, LNCS 576, pp.129-140, Springer-Verlag, 1992.
16. R.Perlman and C.Kaufman, "Secure Password-Based Protocol for Downing a Private Key", *Proc. 1999 Network and Distributed System Security Symposium*, Internet Society, Jan.1999.
17. A.Shamir, "How to Share a Secret", *Communication of the ACM*, Vol.22, No.11, pp.612-613, Nov.1979.
18. D.Vo, F.Zhang and K.Kim "A New Threshold Blind Signature Scheme from Pairings", *SCIS2003*, Vol.1/2, pp.233-238, Jan.2003.
19. T.Wu, "The Secure Remote Password Protocol", *Proc. of Network and Distributed System Security Symposium*, pp.97-111, Internet Society, Jan.1998.
20. T. Pedersen, "Non-interactive and Information theoretic Secure Verifiable Secret Sharing", *Advances in Cryptology - CRYPTO '91*, LNCS 576, pp.129-140, Springer-Verlag, 1992.
21. J. Cha and J. Cheon, "An Identity-Based Signature from Gap Diffie-Hellman Groups", *PKC 2003*, LNCS 2567, pp.18-30, Springer-Verlag, 2003.