Key Generation Protocol in IBC

Dhruti Sharma[1] and Devesh Jinwala[2]
(Corresponding author: Dhruti Sharma)

Information Technology Dept., Sarvajanik College of Engineering & Technology[1]
Dr. R.K. Desai Marg, Athwalines, Surat-1, Gujarat, India.
Department of Computer Engineering, S V National Institute of Technology[2]
Ichchhanath, Surat-7, Gujarat, India.

(Received Jan. 5, 2012; revised and accepted June 12, 2012)

Abstract

Identity Based Cryptography (IBC) is well known method in the field of security, however it has an inherent drawback of Key Escrow under which central key generation center is a sole issuing authority of private key and thus could misuse it. Our survey finds numerous solutions of key escrow; out of them secure key issuing protocol (SKIP) is most prominent. However, there are scopes for improving communication efficiency of this protocol. Thus, we propose an improved version of this protocol by employing clustering approach. With theoretical and empirical analysis, we demonstrate that the proposed approach indeed effectively reduces the communication overheads and improves efficiency of the original secure key issuing protocol.

Keywords: Bilinear pairing, identity based cryptography, key escrow problem, key generation center, key privacy authority

1 Introduction

The traditional public key cryptography (PKC) provides an efficient way of communication using public/private key pair where private key remains secret with the owner while corresponding public key is published in an open directory [7]. As compared to symmetric key cryptography (SKC), PKC is more beneficial as it involves only public key for message transmission; no private key is ever transmitted or shared [20]. However, PKC suffers with the problem of public key authentication i.e. public key should be associated with its user in a trusted manner [7]. In practice, authentication of public key is provided by means of digital certificate which binds user's identity with his public key, however it introduces an additional burden over system in the form of elaborate certificate management [1, 15] which includes certificate invocation, revocation, distribution as well as storage. As a consequence, certificate-based public key cryptosystem requires large amount of storage and computing time [1, 6, 13].

To get rid of issues related to certificate based PKC, in 1984 a novel concept Identity Based Cryptography (IBC) has came into existence [21]. IBC can eliminate the explicit authentication of public key by allowing a user's public key to be derived from his identity information, such as an email address, IP address, etc. As with IBC users could not individually generate their identity based private key, recourse is taken to a trusted party called the Key Generation Centre (KGC). Though IBC has many advantages in the aspect of the key management as compared to traditional public key cryptosystem, it has significant shortcomings with respect to the key privacy issues. Since the KGC is a sole issuing authority of the private key to user, the dreaded key escrow problem occurs. A malicious KGC can decrypt any cipher text and forge the signature for any message, rendering the cryptosystem vulnerable. Hence providing privacy and authenticity are crucial aspects in the IBC. Moreover, the KGC is responsible for transmission of the private key to user that requires a secure channel between them.

To deal with inherent key escrow problem of IBC, several techniques have been proposed [2, 3, 4, 9, 10, 11, 12, 16, 17, 18, 22]. As can be seen in literature, one of most prominent solution of key escrow is secure key issuing protocol (SKIP) [18], however it has several issues related to communication and computation complexity. Also this protocol is vulnerable to collusion attack discussed in [5]. Thus, with this paper, we have proposed a protocol namely Key Generation Protocol with some modifications in SKIP to overcome several limitations of it.

The rest of the paper is organized as follows. Related works on key escrow problem are described in Section 2. Section 3 briefly reviews background concepts on bilinear pairing and ID-based cryptography. A brief overview as well as the detail mechanism of our proposed Key Generation Protocol is given in Section 4. Section 5 represents comparative theoretical and empirical analysis respectively. Finally, we conclude in Section 6.

2 Related Work

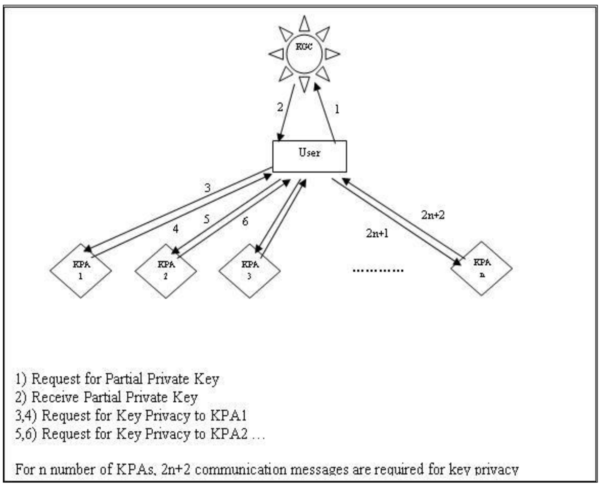
Identity based cryptography (IBC) suffers with the problem of key escrow where central trusted authority issues private key to a user. Since central authority is responsible for private key generation, he is able to work as an authorized user and could be maliciously decipher the incoming encrypted text or could generate false signature. As there are several proposals exist for the solution of key escrow in IBC, they can be easily classified into two groups based on the private key generation techniques: (i) Multiple authority approach, (ii) User chosen secret key information approach. As per our survey, numerous techniques [3, 4, 9, 10, 14, 16, 17, 19] follows multiple authority approach whereas very few [2, 11] are based on secret key information approach.

In multiple authority approach, the critical task of private key generation is distributed among several authorities and hence no single authority could perform any unauthenticated work. Though these methods successfully solve key escrow, they introduce extra overhead on systems and lack of central control of key issuing policy. The user-chosen secret information approaches are either certificate based or certificate less. The certificate based scheme completely overcomes key escrow however since it uses certificate based approach it loses the advantage of an ID based scheme. The certificate less method solves the key escrow problem although it suffers with the problem of public key authentication and so doesn't provide strong security.

As per our literature survey, the most desirable solution of key escrow problem is provided by Secure Key Issuing Protocol (SKIP) [18]. The scheme follows multiple authority approach where single Key Generation Centre (KGC) and multiple Key Privacy Authorities (KPAs) exist. The KGC is responsible for issuing private key to the user whereas KPAs are responsible for key privacy operation. A user computes his private key by first extracting a partial key from KGC and subsequently gets signature of each KPA on this partial key. Though the scheme solves key escrow, it suffers with the problem of large communication overhead as shown in Figure 1.

In addition to this, SKIP is vulnerable to a conspiracy attack discussed in [5] done by KGC and his assistant. If KGC is malicious, then he with his assistant could successfully attack the protocol and obtains an authorized user's private key without colluding with any KPA.

Our studies find several variations [9, 10, 12, 16, 17, 22] of SKIP but no scheme have shown the efforts for improving efficiency of the original version.



1) Request for Partial Private Key
2) Receive Partial Private Key
3,4) Request for Key Privacy to KPA1
5,6) Request for Key Privacy to KPA2 ...

For n number of KPAs, 2n+2 communication messages are required for key privacy

Figure 1: Key privacy in secure key issuing protocol

Some of such variations are proposed in [10, 16] which are based on the concept of threshold cryptography where ***t*** out of ***n*** authorities are involved in user's secrete key computation. Before generating actual private key for a user, all ***n*** authorities have to share secret key which is very critical task. Also as user himself is responsible for the selection of ***t*** authorities, wrong selection may dissipate large amount of time. Even maintenance of large database for all users by all authorities is a significant storage issue.

Other technique, namely Accountable Authority Identity Based Encryption (A-IBE) scheme discussed in paper [12] reduces trust in central authority to solve key escrow. Here, private key can be generated by central private key generator (PKG). The main goal is to restrict the ways in which the PKG can misbehave. But author had remarked that the construction is not very efficient and requires several pairing operations per decryption.

Parallel to A-IBE, Saeran Kwon and Sang-Ho Lee had proposed an identity based key issuing scheme [17] where burden on KGC is reduced by giving the responsibility of user authentication to local registration authority (LRA). A more similar technique was presented in [22]. But in both of the schemes, still collusion attack by dishonest LRA and his assistance is possible. An impostor may misuse the authenticate person's ID and his supported LRA generates his signature on ID. As user ID verification is not done at central KGC, an imposter can get private key of authenticated user. Even private key of user is required to be revoked on daily basis which puts extra burden on entire system.

A somewhat different technique proposed in [23] provides solution of key escrow as well as key exposure. Key insulation and distributing authorities approach is based on the key-insulated mechanism which uses a combination of key splitting and key evolution to protect against key exposure. According to the scheme, the secret key associated with an ID is shared between the user and a physically-secure device. To perform cryptographic operations, the user refreshes his secret key periodically by interacting with the secure device. On the point of efficiency, remark that, in identity-based key-insulated cryptosystem, authentication between user and the physically-secure device is necessary.

3 Background

Since last decades, the concept of Pairing Based Cryptography (PBC) [21] is widely used to design a security system. Basically, PBC includes the usage of different kinds of pairing [3, 15] between elements of two cryptographic groups to third group to construct a secure cryptographic scheme. This section discusses the basic concept of bilinear paring and its usage in identity based cryptography.

3.1 Bilinear Pairing

Bilinear pairing is the mathematical primitive that plays a central role in ID based cryptography. Bilinear map: Let G_1 be an additive group of a prime order $\boldsymbol{q}$ and G_2 be a multiplicative group of the same order. Let P be a generator of G_1. Assume that the Discrete Logarithm Problem (DLP) is hard in both G_1 and G_2 [8, 19].

A Mapping ê: $G_1 \times G_1 \to G_2$ satisfying the following properties are called an *Admissible Bilinear map*.

1. Bilinear: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P,Q \in G_1$ and $a, b \in Z_q^*$.
2. Non-degenerate: ê does not send all pairs of points in $G_1 \times G_1$ to the identity in G_2. (Hence, if P is a generator of G_1 then $\hat{e}(P,P)$ is a generator of G_2).
3. Computable: There exists an efficient algorithm to compute $\hat{e}(P,Q)$ for all $P, Q \in G_1$.

3.2 ID Based Cryptography

The notion of Identity Based Cryptography (IBC) [21] was introduced by Shamir in 1984. The scheme of IBC can be easily designed by means of bilinear pairing. In the basic ID-based scheme [3], a single central trusted authority known as Key Generation Center (KGC) holds a master key s and issues a private key for user based on his ID. The entire mechanism is divided into four basic steps as follows:

Step 1 (Setup): During setup, KGC publishes global system parameters (G_1, G_2, e, H_1, H_2, P_0) where G_1 and G_2 are two cryptographic groups and e is the bilinear map e: $G_1 \times G_1 \to G_2$, H_1: $\{0, 1\}^* \to G_1$ (extract point from ID), H_2: $G_2 \to \{0, 1\}^l$ where l is the length of a plaintext message (hash to the message space) and public key $P_0 = s_0P$ (where secret $s_0 \in Z_q^*$).

Step 2 (Extract): A user extracts private key D_{ID} from KGC through secure channel. KGC computes $D_{ID} = s_0Q_{ID}$ based on the public key $Q_{ID} = H_1(ID)$ where ID is identification of user. User verifies his private key by checking the equality $e(D_{ID}, P)? = e(Q_{ID}, P_0)$.

Step 3 (Encrypt): A sender encrypts message using the public key ID of the receiver. To do this, sender first computes receiver's public key by $Q_{ID} = H_1(ID)$. Then he randomly selects $r \in Zq^*$ and computes the cipher text $C = <U, V>$ where $U = rP$ and $V=m \oplus H_2(e(Q_{ID}, P_0)^r)$. Then C is sent to the receiver user.

Step 4 (Decrypt): Finally, the receiver can decrypt the incoming cipher text $C = <U, V>$ using his private key D_{ID} by computing $V \oplus H_2(e(D_{ID}, U)) = m$. The decryption works because of the bilinear property of the map **e**, i.e. $e(D_{ID}, U) = e(s_0Q_{ID}, rP) = e(Q_{ID}, P_0)^r$.

4 The Proposed Key Generation Protocol

To improve efficiency and to prevent collusion attack of SKIP, we have proposed a Key Generation Protocol in IBC which is based on multiple authority approach. Like SKIP, our protocol includes single KGC and multiple KPAs where KGC issues private key and KPA protects privacy to this private key.

4.1 Overview

Our protocol introduces Net-ID based clustering technique for all available KPAs in system which divides $\boldsymbol{n}$ KPAs into $\boldsymbol{m}$ number of clusters $(m < n)$ as shown in Figure 2. As oppose to SKIP where single system wide public key is exist and it is computed by all existing KPAs cooperatively, in our protocol each cluster has its own cluster public key whose computation involves authorities exist in that cluster only. Each user in a cluster has a private key corresponding to his cluster public key which can be computed by user himself in cooperation with KGC and member KPAs of his cluster. The sender of message has to use receiver's cluster public key for encryption. Decryption can be performed by receiver using his private key.

4.2 Protocol Description

Our proposed protocol works in five phases: System Setup, Public Key Setup, Partial Key Generation, Key Privacy and Private Key Generation. The following are the detail description of each of these phases with their pseudo code.

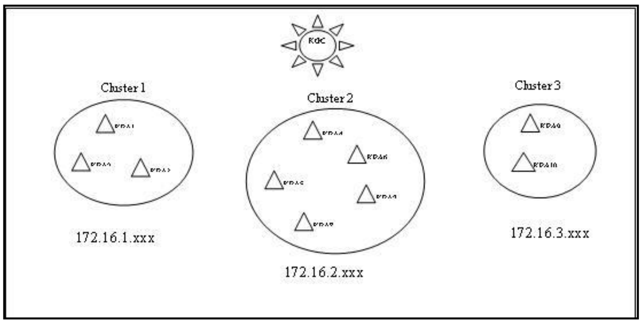


Figure 2: Proposed key generation protocol architecture

1) System Setup: The KGC alone is responsible for doing entire system setup.

Algorithm 1: System Setup

```
Begin
Define two cryptographic groups $G_1$, $G_2$ of some prime
  order q.
Define three hash functions $H_1$, $H_2$, $H_3$ as follows
  $H_1$: $\{0, 1\}^* \to G_1$ (extract point from ID),
  $H_2$: $G_2 \to \{0, 1\}^l$, where l is the length of a plaintext
  message (hash to the message space). $H_3$: Hash onto
  pairing function
Select random master key $s_0 \in Zq^*$.
Compute public key $P_0 = s_0P$.
Publish system parameters ($G_1$, $G_2$, e, $H_1$, $H_2$, $H_3$, $P_0$).
End
```

2) Public Key Setup: As per the architecture of our protocol, all existing KPAs are organized into groups forming clusters. Let us assume a system consists of $\boldsymbol{m}$ clusters and each cluster has different number ($\boldsymbol{k}$) of KPAs. This setup phase contains two steps: (i) cluster public key computation, (ii) public key computation of each KPA. The public key computation for cluster $\boldsymbol{j}$ requires sequential process among all $\boldsymbol{k}$ member KPAs of that cluster.

Algorithm 2: Cluster_PublicKey

```
Begin
Set $Y_{0j} = P_0$.
For each i=1,2,…k KPA in a cluster j
    $Y_{ij}' = s_{ij}Y_{(i-1)j}$
    Send $Y_{ij}'$ to $KPA_{i+1}$
End for
End
```

Algorithm 3: KPA_PublicKey

```
Begin
For each j=1,2,…m clusters
    For each i=1,2,…k KPA in cluster j
        Choose secrete key $s_{ij}$
        Compute Public key $P_{ij} = s_{ij}P$
    End For
End For
End
```

At the end of cluster public key computation, $Y_j \equiv Y_{kj}' = s_{0j}s_{1j}\dots s_{kj}P$ is published as the public key of ***j***th cluster. This public key can be used as a part of encryption by sender of a message to any user in cluster ***j***. Here, the correctness of this sequential process in ***j***th cluster can be verified by $e(Y_{ij}', P) ?= e(Y_{i-1j}', P_{ij})$, where $Y_{0j}' = P_0$.

3) Partial Key Generation: A user with identity (ID) makes request of partial private key to KGC. Here $H_3(e(s_0X; P_0))$ is a blinding factor; a secure channel between the user and the KGC. User can unblind it using his knowledge of x, since $H_3(e(s_0X, P_0)) = H_3(e(s_0xP, P_0)) = H_3(e(P_0, P_0)^x)$.

Algorithm 4: Generate_PartialKey

```
Begin
User (ID) randomly selects x Є Zq*
Compute Blinding Factor X=xP
Send request(X, ID) to KGC.
// KGC side
Verify identity (ID) of a user.
Compute a public key of user from cluster j as
    $Q_{ID} = H_1(ID, KGC, KPA_1, \dots, KPA_k)$
Computes a blinded partial private key as
    $Q_0' = H_3(e(s_0X, P_0))s_0Q_{ID}$
Computes KGC's signature on $Q_0'$ as $Sig_0(Q_0) = s_0Q_0'$
Send $Q_0'$ and $Sig_0(Q_0)$ to the user.
End
```

4) Key Privacy: Key privacy service is performed by user in cooperation with member KPAs of his cluster. Each member KPA provides key privacy by generating partial key based on previous key he received from user in sequential manner. Finally, a user receives $Q_k' = H_3(e(s_kX; P_k))s_kQ_{k-1}$. Subsequently, the user generates his private key using Q_k'.

Algorithm 5: Key_Privacy

```
Begin
For each i=1, 2,…k KPA in cluster j
User send (ID, X, $Q_{i-1}'$ and $Sig_{i-1}(Q_{i-1}')$) to $KPA_i$
//Now $KPA_i$
    Verify the identification of user.
    Checks $e(Sig_{i-1}(Q_{i-1}'); P) ?= e(Q_{i-1}'; P_{i-1})$.
    Computes $Q_i' = H_3(e(s_iX; P_i))s_iQ_{i-1}'$ and
    $Sig_i(Q_i') = s_iQ_i'$.
    Sends $Q_i'$ and $Sig_i(Q_i')$ to the user
End for
End
```

5) Private Key Generation: The user retrieves his private key D_{ID} by unblinding Q_k' in the following way

$$\begin{aligned} D_{ID} &= Q_k' / H_3(e(P_0; P_0)^x) \dots H_3(e(P_k; P_k)^x) \\ &= s_0s_1 \dots s_kQ_{ID} \end{aligned}$$

Algorithm 6: Generate_PrivateKey

```
Begin
For each i=0, 2,…k
    Compute $D_{ID} = Q_k' / H_3(e(P_i; P_i)^x)$
End for
End
```

The user from cluster ***j*** can verify the correctness of his private key by $e(D_{ID}; P)? = e(Q_{ID}; Y_j)$.

Encryption & Decryption

(i) Encryption: To perform encryption a sender needs receiver's public key as well as some global system public key.

Algorithm 7: Encryption

```
Begin
Get the information about cluster of receiver user from
KGC.
Generate receiver's public key by
    $Q_{ID} = H_1(ID, KGC, KPA_1, \dots, KPA_k)$
Select random r Є $Z_q$* and compute
    $C = (U, V) = (rP, m \oplus H_2(e(Q_{ID}; Y_j)^r))$
Send C to the receiver users
End
```

(ii) Decryption: To decrypt the incoming cipher text, a receiver uses his private key D_{ID} as follows.

Algorithm 8: Decryption

```
Begin
Compute $V \oplus H_2(e(D_{ID}, U)) = m$.
End
```

5 Analysis

This section entirely discusses theoretical as well as empirical analysis of our protocol. Section 5.1 describes theoretical evaluation of our protocol as compared to SKIP and other protocols. Furthermore, as mentioned in Section 1, our prime concern is to improve efficiency of SKIP, each of the resultant tables and corresponding figures in Section 5.2, is showing the comparative analysis of our protocol with original SKIP only.

5.1 Theoretical Analysis

As discussed in Section 4, our proposed protocol uses clustering approach whereby it divides a system with ***n*** KPAs into ***m*** clusters. Each cluster C_i (where i=1, 2, .., m) has different member KPAs K_i where $K_i < \boldsymbol{n}$. A private key for user in cluster C_i is computed by KGC and member KPAs of his cluster. Thus a cluster having total K_i KPAs needs $2K_i+2$ messages for private key computation which is excessively less than 2n+2 communication message of SKIP as shown in Figure 1. Moreover, our protocol proposes simultaneous computation of cluster public key, as each cluster has different member KPAs and no KPA is shared among clusters. So if t_m is the time needed by largest cluster (having maximum KPAs) for public key computation, all other clusters could compute their public key in time <= t_m because all of them contain less number of KPAs than largest cluster. It means total time (time needed for public key computation of all clusters) is equal to time required by the largest cluster (t_m) to compute its public key. In contrast to this, SKIP suffers from the problem of large public key computation time as all authorities are participated in it.

Also our protocol requires less number of communication messages for public/private key computation as compared to [10, 16, 22] since all these schemes use threshold cryptography for private key generation which needs sharing of secrete key. This secrete sharing involves all ***n*** existing KPAs. Moreover encryption in all these schemes is based on system public key whose calculation requires participation of all KPAs and hence needs more number of message exchanges as compared to our protocol.

So, this entire discussion proves that our protocol effectively uses less communication messages and hence reduces computation time which were the major limitations of SKIP [18] and other related schemes [10, 16, 22].

In addition to this, our protocol offers more reliability as compared to all existed multiple authority schemes [3, 4, 9, 10, 14, 16, 18, 19, 22] as they involve all authorities in public/private key computation and so failure of one entity will break the entire system. In contrast to this, in our protocol, failure of any KPA affects the computation of public key of that cluster and computation of private key of all users in that cluster only. Public key of other clusters as well as private key of users in other clusters could be easily computed without any interruption.

Moreover, all [3, 4, 9, 10, 14, 16, 18, 19, 22] schemes require multiple same kinds of authorities which are responsible for user registration and verification. As a result higher the number of authorities; higher is infrastructural overhead in the system. Furthermore, all users in system have to register with each authority which puts additional burden on user also. Even each authority is responsible for each user ID verification so large database maintenance and hence excess infrastructural overhead is a major issue in all above systems. As compared to this, our scheme provides more convenient solution of multiple authority approach whereby a single central KGC is responsible for registration of all users in the system and KPAs are in charge of local users in his cluster only.

Another major advantage of our protocol is it prevents the collusion attack [5] caused by malicious KGC and his assistant which was the major issue in SKIP.

Proof: Let assume a malicious KGC with his assistant (unauthorized user) is trying to generate private key of an authorized user based on his ID. So formally, an assistant is asking for partial key to KGC and as KGC is malicious, it provides blinded partial key to him. With this partial key, an assistant is trying to perform key privacy operation (step 4) in cooperation with all KPAs in its own cluster. But as per step 4 of our protocol, each KPA (before signing onto partial key), it first verifies identity of user which was not part of key securing step of SKIP. If requested entity is not a valid user, KPA could grab his dishonest activity. Thus no user or even KGC can illegally generate private key of an authorized user. In this way, our scheme provides strong security against any collusion attack done by KGC.

Finally we can claim that our proposed protocol is an efficient and a complete solution of key escrow problem. The lacking part of our approach is that each KPA is responsible for ID verification however it doesn't overburden the system like other scheme [3, 4, 9, 10, 14, 16, 18, 19, 22] as in our protocol each authority is not responsible for identity verification for all users in the system. In addition, the size of cluster is still a research issue although our analysis defends that the cluster size should not be too small so it loses the advantage of clustered approach. Generally, it depends upon the real application.

5.2 Empirical Analysis

To justify the above theoretical analysis, we have tested our protocol for different number of KPAs in a system with different sized clusters. We have used simulation time, storage and energy as the evaluation matrices for our proposed algorithm. The two major phases of our protocol which consumes large part of total execution time are public key setup (Phase 2) and key privacy (Phase 4). Thus entire empirical analysis represents all the results in tabular as well as graphical format for both of these phases.

5.2.1 Timing Analysis

As discussed in Section 4.1, our key generation protocol employs clustering approach for organization of KPAs. It divides a system containing ***n*** KPAs into ***m*** (m<n) clusters and each cluster has ***k*** member KPAs. All clusters could compute their public key in parallel manner as no KPAs are shared among clusters. In addition the preprocessing for private key generation i.e. key privacy phase involves only member KPAs of user cluster. Thus, it is clear that both public/private key computation needs less amount of message exchanges as compared to SKIP and so can be simulated in lesser time than SKIP. The Table 1 and Table 2 show total simulation time for public/private key computation.

Table 1: Simulation time for public key setup

No. of nodes (n)	**No. Of KPAs**	**Time requirement (ms)**				
		SKIP	***No. of clusters in Proposed Key Generation Protocol***			
			2	***3***	***4***	***5***
13	10	352	192	128	96	64
20	17	608	320	224	192	128
25	22	800	384	288	224	192
30	27	960	480	320	256	224
35	32	1120	576	384	288	256
40	37	1312	672	480	352	320

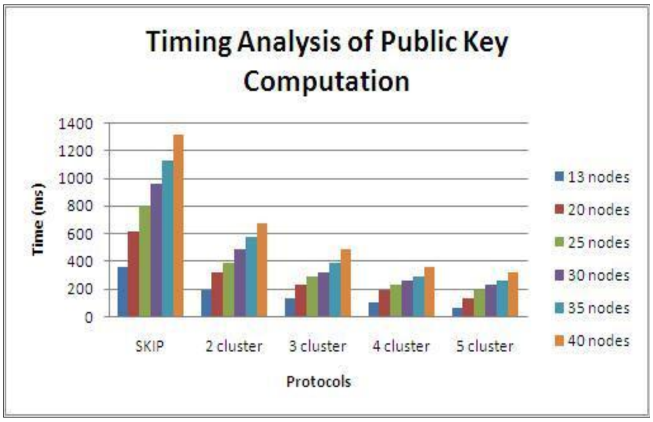


Figure 3: Timing analysis for public key setup

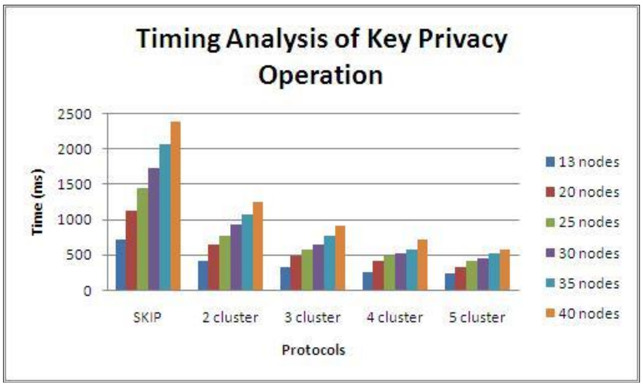


Figure 4: Timing analysis for key privacy

Table 2: Simulation time for key privacy

No. of nodes (n)	**No. Of KPAs**	**Time requirement (ms)**				
		SKIP	***No. of clusters in Proposed Key Generation Protocol***			
			2	***3***	***4***	***5***
13	10	704	416	320	256	224
20	17	1120	640	480	416	320
25	22	1440	768	576	480	416
30	27	1728	928	640	512	448
35	32	2048	1056	768	576	512
40	37	2368	1248	896	704	576

The graphical results in Figure 3 and Figure 4 shows that our proposed protocol with 2-clusters reduces cluster public key computation time by almost 45 - 50% and private key computation time by more than 40% as compared to the original SKIP. Thus total simulation time of our protocol is almost half of the simulation time required by SKIP. Furthermore increase in number of clusters, decreases required computation time. Formally, assume a system with ***n*** KPAs is supposed to take t_n time unit for execution of entire SKIP. If the same system follows our proposed protocol with 2-clusters approach, it will take $\Theta(t_n/2)$ time unit. If it follows 3-clusters approach, it will take $\Theta(t_n/3)$ and so on. In general, we can say that our protocol completes the execution of entire protocol for the system with ***k*** clusters within $\Theta(t_n/k)$ time unit which is k-times less as compared to time taken by SKIP.

Table 3: Usage of RAM

No. of nodes (n)	**No. Of KPAs**	**RAM usage (MB)**				
		SKIP	***No. of clusters in Proposed Key Generation Protocol***			
			2	***3***	***4***	***5***
13	10	3.41	3.51	3.61	3.70	3.76
20	17	4.73	4.83	4.88	4.98	5.03
25	22	5.64	5.74	5.79	5.89	5.94
30	27	6.55	6.65	6.71	6.76	6.85
35	32	7.46	7.56	7.62	7.71	7.76
40	37	8.37	8.47	8.53	8.62	8.68

5.2.2 Storage Analysis

The proposed key generation protocol uses cluster based architecture in which each cluster has its own set of KPAs. As we have already discussed, each cluster has its own public key. So if a system has total ***m*** separate clusters, then ***m*** different public keys also exist in the system which requires somewhat more storage space as compared to SKIP. Table 3 and Table 4 show our experimental results of RAM and ROM usage.

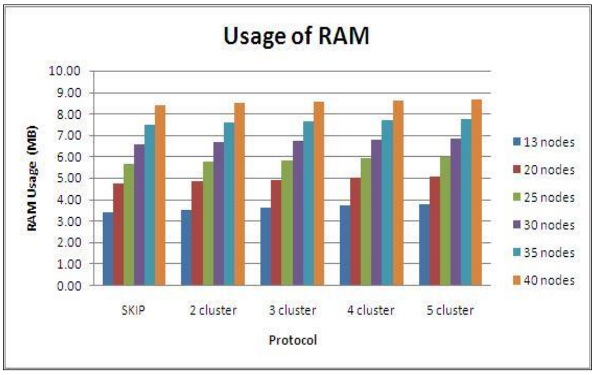


Figure 5: Usage of RAM

Table 4: Usage of ROM

No. of nodes (n)	No. Of KPAs	ROM usage (MB)				
		SKIP	*No. of clusters in Proposed Key Generation Protocol*			
			2	*3*	*4*	*5*
13	10	83.0	87.0	88.5	92.5	96.5
20	17	82.0	84.5	88.5	92.5	96.5
25	22	82.0	85.0	88.5	92.5	96.5
30	27	81.5	84.5	88.5	92.0	96.5
35	32	82.0	84.5	88.0	92.5	96.5
40	37	81.5	84.5	88.0	92.5	96.5

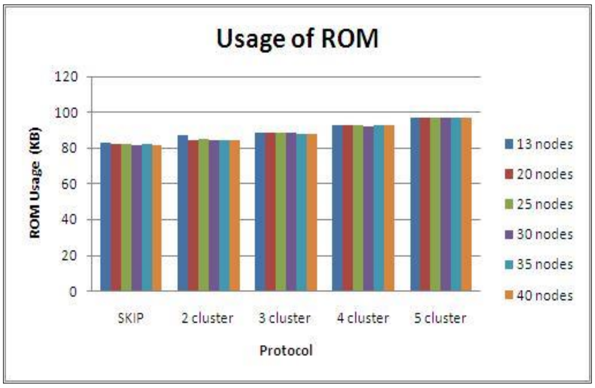


Figure 6: Usage of ROM

The graphical results in Figure 5 and Figure 6 demonstrate that the deployment of 2-clusters in system needs very little extra storage i.e. RAM and ROM. As we increase number of clusters, the requirement of RAM/ROM is rising slightly. Our results prove that the proposed scheme requires approximately 3-15% more storage space as compared to SKIP however the reduction in computation and communication time conceals this small overhead introduced in the system.

5.2.3 Energy Analysis

As per our proposed architecture, cluster public key computation is a separate activity performed by member KPAs of that cluster only. Generation of public key of one cluster doesn't intertwining into computation of others. Thus, all clusters in the system could simultaneously perform their public key computation. This helps in improving system throughput, throughput per node as well as radio propagation. The results given in Table 5 and Table 6 show the total throughput of entire system during public key setup phase and key privacy phase respectively. The corresponding results are shown in Figure 7 and Figure 8. The results for throughput per node are given by Table 7 and Table 8. Again the graphical view of result is shown by Figure 9 and Figure 10. Table 9 shows average radio analysis of our algorithm as compared to SKIP and it is graphically represented by Figure 11.

Table 5: Total system throughput during public key setup

No. of nodes (n)	No. Of KPAs	Throughput (MHz)				
		SKIP	*No. of clusters in Proposed Key Generation Protocol*			
			2	*3*	*4*	*5*
13	10	160.59	167.88	186.93	192.96	201.06
20	17	185.47	219.77	215.79	221.22	245.51
25	22	202.78	206.05	214.64	215.03	216.17
30	27	204.75	208.88	208.94	209.00	209.81
35	32	189.66	200.91	210.82	211.04	213.77

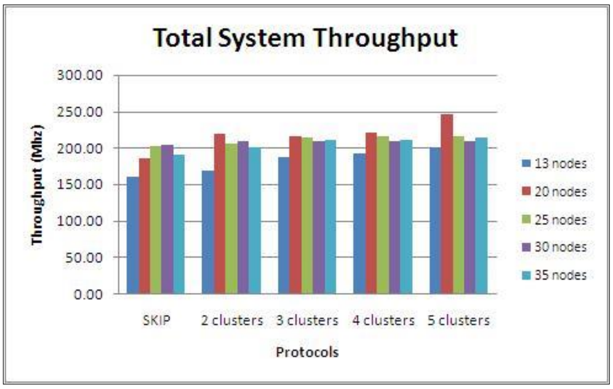


Figure 7: System throughput during public key setup

Table 6: Total system throughput during key privacy

No. of node (n)	No. Of KPAs	Throughput (MHz)				
		SKIP	*No. of clusters in Proposed Key Generation Protocol*			
			2	*3*	*4*	*5*
13	10	194.52	194.22	204.24	210.28	210.77
20	17	207.95	209.73	211.63	212.12	221.60
25	22	173.04	177.29	181.56	186.33	192.89
30	27	186.94	202.72	208.07	209.85	213.51
35	32	186.12	191.93	198.92	209.27	212.90

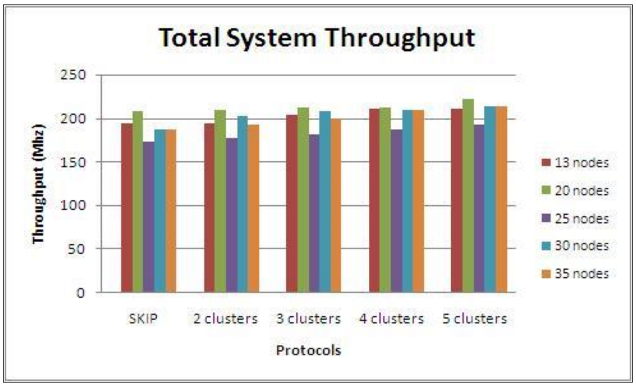


Figure 8: System throughput during key privacy phase

Table 7: Throughput per node during public key setup

No. of node (n)	No. Of KPAs	Throughput (MHz)				
		SKIP	*No. of clusters in Proposed Key Generation Protocol*			
			2	*3*	*4*	*5*
13	10	12.35	12.86	14.38	14.84	15.47
20	17	9.27	10.99	10.79	11.06	12.28
25	22	8.28	8.24	8.54	8.56	8.61
30	27	6.39	6.47	6.86	6.91	8.39
35	32	5.42	5.74	5.95	6.03	6.11

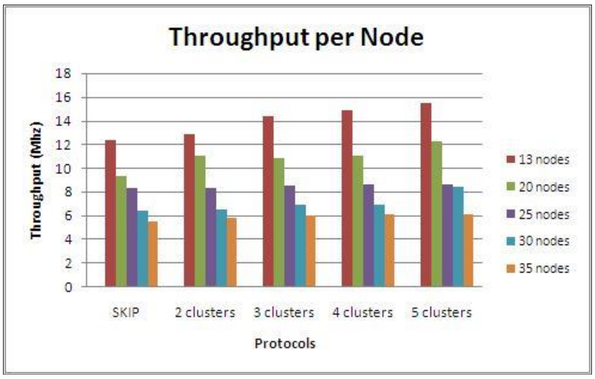


Figure 9: Throughput per node during public key setup

As discussed in Section 4.1, our clustering approach involves less number of authorities in public/private key calculation; it has fewer amounts of communication and computation and thus increases system throughput and throughput per node as compared to SKIP. The results in Figure 8 to Figure 11 demonstrate that, by increasing the number of clusters we can effectively generate 7-20% more throughputs as compared to SKIP.

Table 8: Throughput per node during key privacy

No. of node (n)	No. Of KPAs	Throughput (MHz)				
		SKIP	*No. of clusters in Proposed Key Generation Protocol*			
			2	*3*	*4*	*5*
13	10	14.96	14.94	15.71	16.18	16.21
20	17	10.40	10.49	10.58	10.61	11.08
25	22	6.92	7.09	7.26	7.45	7.72
30	27	6.23	6.64	6.94	6.99	7.12
35	32	5.32	5.48	5.68	5.98	6.08

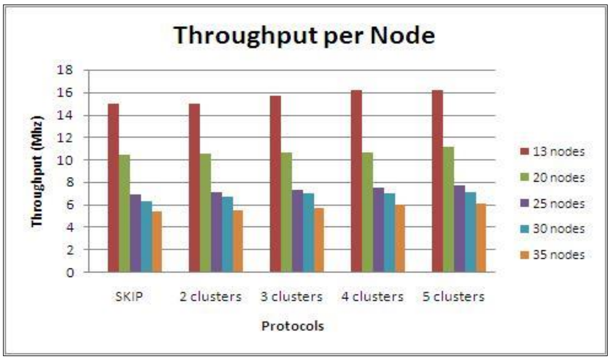
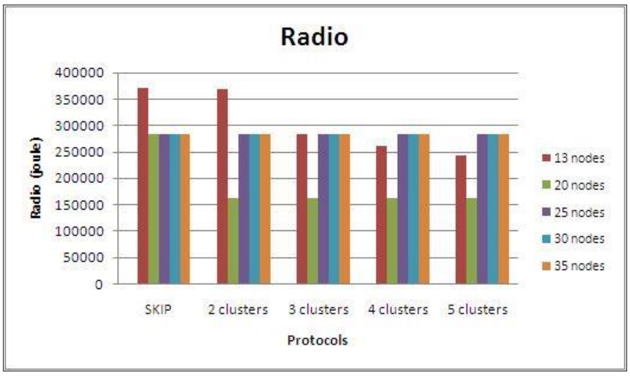


Figure 10: Throughput per node during key privacy

We get effective deduction in average propagation as shown in Figure 11 as proposed clustering approach decreases communication messages. From all the above empirical analysis, we can state that our proposed key generation protocol is more efficient than SKIP in terms of communication and computation.

Table 9: Average radio analysis



No. of nodes (n)	No. Of KPAs	Radio (K Joule)				
		SKIP	*No. of clusters in Proposed Key Generation Protocol*			
			2	*3*	*4*	*5*
13	10	370	369	283	260	243
20	17	283	161	162	162	162
25	22	283	283	283	283	283
30	27	283	283	283	283	283
35	32	283	283	283	283	283

Figure 11: Average radio analysis

6 Conclusion

As discussed, key escrow is the major issue associated with the private key generation in identity based cryptography. To resolve this problem, several mechanisms are available however as per our literature survey no one is completely satisfactory. Through this paper, we have proposed a new key generation protocol which is based on the existing Secure Key Generation Protocol (SKIP). The proposed protocol improves efficiency of SKIP and thus solves some of the practical problems associated with SKIP. It uses cluster based approach for organization of multiple authorities exists in the system which produces several separate clusters. Our theoretical analysis shows that the proposed clustering approach decreases the amount of communication messages in public/private key computation and thus reduces total simulation time and improves the system throughput. Our empirical analysis proves this by showing as we increase the number of clusters, the simulation time of our protocol is going to reduce and corresponding throughput is going to increase. Moreover, as per the detailed protocol description, all the authorities perform user identity verification which provides protection against conspiracy attack done by malicious KGC. By presenting our results in tabular and graphical format, we have come to conclusion that our protocol provides a complete, efficient and best-of-breed solution of the key escrow problem- an inherent drawback of IBC.

Acknowledgments

This study was supported by the Sardar Vallabhbhai National Institute of Technology, Surat (India). The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] C. Adams and S. Lloyd, Understanding Public-Key Infrastructure – Concepts, Standards, and Deployment Considerations. Macmillan Technical Publishing, Indianapolis, USA, 1999.

[2] S. Al-Riyami and K. Paterson, “Certificateless public key cryptography a full version,” A short version appeared at Asiacrypt 2003, LNCS 2894, pp. 452-473, Springer-Verlag, 2003.

[3] D. Boneh and M. Franklin, “Identity-based encryption from the Weil pairing,” Advances in Cryptology (Crypto 2001), LNCS 2139, pp.213-229, Springer-Verlag, 2001.

[4] L. Chen, K. Harrison, N. Smart and D. Soldera, “Applications of multiple trust authorities in pairing based cryptosystems,” International Conference on Infrastructure Security (InfraSec 2002), LNCS 2437, pp.260-275, Springer-Verlag, 2002.

[5] X. Chunxiang, Z. Junhui, and Z. Qin, “A note on secure key issuing in ID-based cryptography,” Cryptology ePrint Archive, Report 2005/180 (http://eprint.iacr.org/2005/180.pdf), 2005.

[6] J. Dankers, T. Garefalakis, R. Schaffelhofer, and T. Wright, “Public key infrastructure in mobile systems,” IEEE Electronics and Communication Engineering Journal, Vol. 14, pp. 180–190, 2002.

[7] W. Diffie and M. Hellman, “New directions in cryptography,” IEEE Transactions on Information Theory, Vol. IT-22, pp. 644–654, Nov. 1976.

[8] S. D. Galbrahaith, K. Harrison, and D. Soldera, “Implementing the tate pairing, algorithmic number theory,” 5th International Symposium, ANTS-V, LNCS 2369, pp. 324–337, Springer-Verlag, 2002.

[9] R. Gangishetti, M. C. Gorantla, M. L. Das, A. Saxena and V. P. Gulati, “An efficient secure key issuing protocol in ID-based cryptosystems”, In Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC 2005), Volume-1, IEEE Computer Society, 4-6 April, 2005, Las Vegas, USA, pp. 674-678.

[10] R. Gangishetti, M. C. Gorantla, M. Das, and A. Saxena, “Threshold key issuing in identity-based cryptosystems”, Computer Standards & Interfaces, Vol. 29, no. 2, pp. 260–264, 2007.

[11] C. Gentry, “Certificate-based encryption and the certificate revocation problem,” Advances in Cryptology(EUROCRYPT 2003), LNCS. 2656, pp. 272-293, Springer-Verlag, 2003.

[12] V. Goyal, “Reducing trust in the PKG in identity based cryptosystems”, In Advances in Cryptology (CRYPTO’07), pp. 430–447, 2007.

[13] P. Gutmann, “PKI: It’s not dead, just resting,” IEEE Computer, Vol. 35, no. 8, pp. 41–49, 2002.

[14] F. Hess, “Efficient identity based signature schemes based on pairings,” Selected Areas in cryptography (SAC 2002), LNCS 2595, pp. 310–324, Springer-Verlag, 2002.

[15] M. Joye and G. Neven, “Identity-based cryptography,” Cryptography and Information Security Series, IOS Press, Vol.2, 2009.

[16] K. P. Kumar, G. Shailaja, A. Saxena, “Secure and efficient threshold key issuing protocol for ID-based cryptosystems”, Cryptology ePrint 2006/245, 2006.

[17] S. Kwon, S. H. Lee, “Identity-based key issuing without secure channel in a broad area”, Information Security Applications, LNCS 2007, 4298, pp. 30-44, Springer-Verlag, 2007.

[18] B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang and S. Yoo, “Secure key issuing in ID-based cryptography,” ACM Second Australasian Information Security Workshop, pp.69-74, New Zealand, 2004.

[19] K. G. Paterson, “Cryptography from pairings, a snapshot of current research,” Information Security Technical Report, Vol. 7(3), pp. 41–54, 2002.

[20] RSA Laboratories, “What is public-key cryptography?” online : http://www.rsa.com/rsalabs/

[21] A. Shamir, “Identity-based cryptosystems and signature schemes,” Advances in Cryptology (Crypto 1984), LNCS 196, pp.47-53, Springer- Verlag, 1984.

[22] C. Wang, J. Liu, “A practical key issuing scheme in identity-based cryptosystem”, ISECS International Colloquium on Computing, Communication, Control, and Management, 2008. CCCM '08., Volume. 3, 2008.

[23] J. Wang, X. Bai, J. Yu, and D. Li, “Protecting against key escrow and key exposure in identity-based cryptosystem”, TAMC 2007, LNCS 4484, pp. 148–158, Springer-Verlag 2007.

Ms. Dhruti Sharma is serving as an Assistant Professor in Information Technology Department with Sarvajanik College of Engineering and Technology, Surat(India). Her major areas of interests are Information security, Wireless sensor network and Distributes systems.

Dr. Devesh Jinwala is serving as an Associate Professor in Computer Engineering with Sardar Vallabhbhai National Institute of Technology, Surat (India). His major research areas of interests are Information Security in general and that in Resource Constrained Environments, specifically; Algorithms & Computational Complexity and Using Ontologies in Software Requirements and Specifications.