Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 46 – 57

doi:10.3906/elk-1407-213

Comparison of group key establishment protocols

Serap ŞAHİN*, Rabia ASLANOĞLU
Department of Computer Engineering, Faculty of Engineering, İzmir Institute of Technology, İzmir, Turkey

Received: 04.08.2014 • **Accepted/Published Online:** 13.11.2015 • **Final Version:** 24.01.2017

Abstract: Recently group-oriented applications over unsecure open networks such as Internet or wireless networks have become very popular. Thus, group communication security over unsecure open networks has become a vital concern. Group key establishment (GKE) protocols are used to satisfy the confidentiality requirement of a newly started communication session by the generation or sharing of an ephemeral common key between the group members. In this study, we analyze the computation and communication efficiency of GKE protocols. Besides confidentiality, the security characteristics of identification and integrity control are also required for all steps of the protocol implementations. Thus, the main contribution of this work is to provide the computation and communication efficiency analysis of the same GKE protocols along with the identification of the group entities and integrity control of messages during the protocol steps. The specific implementation and analysis of GKE protocols are performed by group key agreement (GKA) with pairing-based cryptography and group key distribution (GKD) with verifiable secret sharing, respectively. Finally, a comparison of GKA and GKD protocols on the basis of their strong points and cost characteristics are also provided to inform potential users.

Key words: Group key establishment, secure communication, pairing-based cryptography, verifiable secret sharing.

1. Introduction to group key establishment protocols

The establishment of a common secret key between the members of a group over an open unsecure network is vital to satisfy secure communication requirements. The confidentiality of the communication between members of a group requires sharing a common ephemeral key to use in the encryption and decryption of communication data. For this purpose, there are two group key establishment (GKE) protocols: group key agreement (GKA), a protocol whereby the parties jointly establish a common secret, and group key distribution (GKD), a protocol whereby one of the parties creates or obtains a secret value and then securely distributes it to other parties.

The first security drawback of both methods is the lack of authentication. For this reason, an active adversary can compromise the communication of legitimate parties by the use of a "man in the middle attack". The second security drawback is the lack of confidentiality and integrity checking during the protocol execution.

Authentication is generally based on long-term keys, which can be associated with identities. There are two main approaches to provide authentication: public key infrastructure (PKI) and identity based (ID-based) infrastructure. However, ID-based cryptography has two important disadvantages. First, it suffers from the key escrow problem since the private key generator generates private keys of entities using its master secret key. Second, it requires a secure channel to distribute generated private keys to participants. In this study:

*Correspondence: serapsahin@iyte.edu.tr

(i) The GKA original protocol model is based on Lin et al.'s multiparty key agreement protocol [1]. Our contribution is the usage of a digital signature in order to provide message integrity checking and authentication between the users with a PKI solution.

(ii) The GKD original protocol model is based on Feldman's verifiable secret sharing (VSS) key distribution protocol [2,3]. Due to nature of the VSS protocol, there is a secure channel requirement. We propose the usage of PKI, encryption, and signature schemes, which are realized in the execution of protocol over unsecure public networks.

In this article, we analyze and compare the communication and computation costs of these two GKE protocol models with and without our contribution (i.e. authentication and integrity checking).

Section 2 presents the notations and specifications of the implemented protocols. In Section 3, the construction of GKA and GKD protocols is examined with their implementation results and security and efficiency analyses with and without our contributions. Finally, Section 4 presents the results of the efficiency analysis and implementation of the GKE protocols.

2. Notations, assumptions, and specifications of implemented protocol models

2.1. Notations used in the implementation of protocol models

The common notations of both protocol implementations are:

- n is the number of participants in the group.
- U_i is a participant of group, $U_i \in \{U_1, U_2, \ldots, U_n\,,\ i = 1, .., n\,.$
- $Cert_i = \{ID_{i,}\ Y_i$ is the certificate of U_i and its long-term public key is Y_i, and the unique numeric identifier is $ID_{i,}\,.\ Cert_i$ is signed by the certification authority (CA).
- H is an SHA-512 secure hash function.

The specific notations of key agreement protocol are:

- A bilinear map $e : G_p \times G_p \rightarrow G_T$ between the two groups G_p and G_T as long as a variant of the computational Diffie–Hellman assumption is hard on $G_p\,.$
- p is the prime number.
- P is the generator of additive group $G_p\,.$
- K is the common secret key.
- a_i is the long-term private key randomly chosen by $U_i\,.$
- Y_i is the long-term public key computed $Y_i = a_i P$ by the related party $U_i\,.$
- x_i is the short-term (ephemeral) secret key randomly chosen by $U_i\,.$
- T_i and X_i are public messages.
- Public parameters are $param = \langle G_p, G_T, e, p, P, H\rangle\,.$

The specific notations of key distribution protocol are:

- Primes $q$and $p$such that $q \,|\, (p-1)$, and a generator $g \in Z_p^*$ is an element of order q.
- s is common secret that will be shared.
- $s_i \in \{ s_1, s_2, \ldots, s_n \}$ represent subsecrets.
- L is a leader who shares the common secret s among the n participants.
- x_i is U_i's long-term secret key of corresponding public key Y_i.
- t is the threshold value, according to Shamir's (t, n)secret sharing scheme.
- $f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_t x^{t-1}$, is a $(t-1)$ degree polynomial in which secret $s = a_0$ and $[\, a_0, \ldots, a_t \,] \in Z_q$.
- C is the commitment vector including different t DLog commitments;

$$C_i = g^{a_i},\ i = 0, .., t\,.$$

- $(\, sig_1 sig_2 \,)$ is signature pair and $\left(Enc_i^1, Enc_i^2 \right)$ is an ElGamal encryption pair.
- Public parameters are $param = \langle q, p, g, H \rangle$.

2.2. Assumptions used in the implementation of protocols

- Before the protocol starts, every party U_i generates his/her $Cert_i = \{ ID_i, Y_i \}$ from a CA and these certificates of users have already been exchanged between group participants.
- The protocol works over an open unsecure channel forming a secure, closed communication group of participants. The members of the group are called reliable; they do not leak secret information outside of the group, and they send the correct messages defined by the protocol. All participants agree to follow the protocol.

2.3. Specifications of the implemented protocols

For key agreement protocol:

- The protocol models use the broadcast communication model.
- For each run of the protocol, there is one among n parties, which works as a leader L and starts the protocol specifying the static group number.

For key distribution protocol:

- The protocol needs two constant rounds, namely sharing and reconstruction, with n parties who participate in the establishment of the common secret and a leader L who shares the common secret s and commitments C among the n participants.
- There is a threshold value, which means that more than t shared subsecret values can be used to reconstruct the common secret generated by the leader. The relation between the threshold t and n is $\geq (\, 2t+1 \,)$ [4,5].

3. Implementation of GKA and GKD protocols

A more detailed documentation and numeric examples of these implementations were presented in [6].

3.1. The common implementation environment

Both algorithms are implemented for a small group of users as U_i, $i = 1, .., 4$. ANSI C is preferred as the programming language and the GNU Multiple Precision Arithmetic (GMP) Library (http://gmplib.org/) is chosen for multiple precision calculations, while the OpenSSL Library (http://www.openssl.org/) is used for SHA-512 (http://csrc.nist.gov/publications/fips). The Pairing-Based Cryptography (PBC) Library (http://crypto.stanford.edu/pbc/) is used for the pairing map and elliptic curve calculations are used for the key agreement protocol. The configuration of the machine is an Intel Core 2 Duo 2.0 GHz CPU, 4 GB RAM, and Ubuntu 12.10 operating system.

3.2. Security contributions to the GKA protocol

In this study, the communication among reliable participants satisfies the confidentiality via pairing-based cryptography and identification, integrity, and nonrepudiation via the Boneh–Lynn–Shacham (BLS) short signature scheme [7,8].

3.3. Security contributions to the GKD protocol

The protocol is transformed into a feasible form to apply over an open-unsecure channel by adding encryption and signature algorithms. In this study, the communication between reliable participants satisfies the confidentiality via cryptographic structures such as the ElGamal encryption algorithm and satisfies the identification, integrity, and nonrepudiation via the ElGamal signature scheme [9]. To improve the security level of the implementation, ElGamal encryption and signature schemes can be implemented by different group structures or schemes as indicated in [10,11].

3.4. Implementation and efficiency analysis of GKA protocol

Setup: This is an offline task and numeric values of input and setup parameters are calculated by use of the PBC Library. E is a super singular curve defined by $y^2 = x^3 + x$ over F_q and $P \in E/F_q$ is a 1024-bit generator. The Weil pairing on the curve E / F_{q^2} is a mapping: $G_q \times G_q \rightarrow G_T$. The $q = p \bullet h + 1$ is 512-bit prime number where h is a multiple of 12 and p is 160-bit prime number. Each U_i should have static public Y_i as a 1024-bit and private a_i as a 160-bit key pair where $Y_i = a_i P$.

Now the protocol can be started by leader L. Here the process of the protocol is given in two sections as "GKA - protocol steps" and "GKA - protocol steps with contributions" as depicted in Figures 1 and 2, respectively.

3.4.1. GKA protocol steps as depicted in Figure 1

The leader L from set U announces the public parameters $param = \langle G_p, G_T, e, p, P, H \rangle$ and the group number n. Then the first round starts.

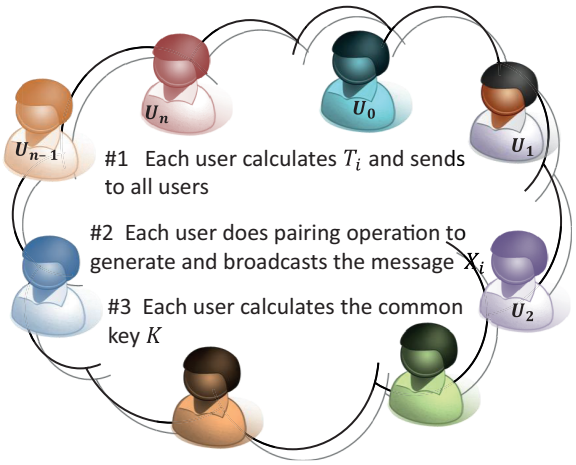

Figure 1. GKA - protocol steps.

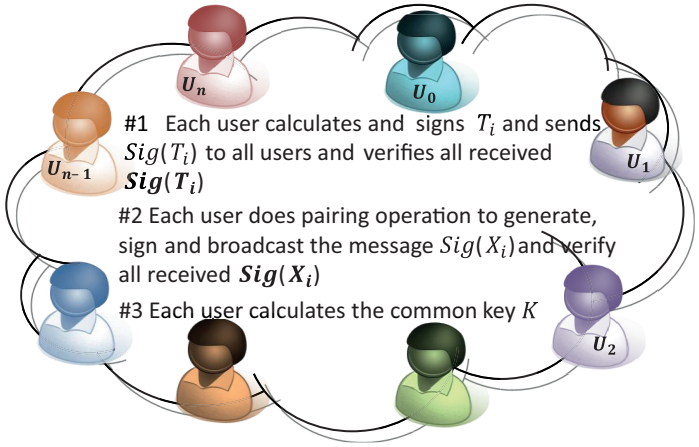

Figure 2. GKA - protocol steps with contributions.

Round 1 - Step 1: Each user U_i, $i = 1, \ldots, n$, chooses a random ephemeral secret number x_i, computes $T_i = x_i Y_i = x_i(a_i P)$, and sends T_i to all users.

Round 2 - Step 2: Each user does pairing operation to generate and broadcast the message $X_i = e\left((Y_{i+1}+T_{i+1}),\ (Y_{i+2}+T_{i+2}) - (Y_{i-1}+T_{i-1})\right)^{(a_i + a_i x_i)}$.

Round 2 - Step 3: Each user calculates the common key:

$$K_i = e\left((Y_{i+1}+T_{i+1}),\ n \times (Y_{i-1}+T_{i-1})\right)^{a_i + a_i x_i} \times X_i^{n-1} \times X_{i+1}^{n-2} \times \ldots \times X_{i-2} =$$

$$e(P,P)\left[\begin{array}{l}(a_n + a_n x_n) \times (a_1 + a_1 x_1) \times (a_2 + a_2 x_2) + (a_1 + a_1 x_1) \times (a_2 + a_2 x_2) \times (a_3 + a_3 x_3) + \ldots \\ + (a_{n-1} + a_{n-1} x_{n-1}) \times (a_n + a_n x_n) \times (a_{n+1} + a_{n+1} x_{n+1})\end{array}\right].$$

Furthermore, the common shared secret key is then obtained by the key derivation function algorithm- kdf, which is simultaneously used to calculate the common secret by each U_i (http://www.emc.com/emc-plus/rsa-labs/standards-initiatives/pkcs-5-password-based-cryptography-standard.html).

$$K = kdf(\ K_i \parallel U_1 \parallel U_2 \parallel U_3 \parallel U_4\ ).$$

3.4.2. GKA protocol steps with contributions as depicted in Figure 2

Round 1 - Step 1: Each user calculates and signs T_i and sends $Sig_i(T_i)$ to all users for identification and integrity checking, and at the same time verifies all received $Sig_i(T_i)$. In implementation, each user generates a digest value of T_i by $H_i = SHA-512(T_i)$. Then U_i signs the H_i value with his static private key a_i by using the BLS short signature scheme. Here the BLS output is a point on E/F_q with 1024-bit length.

Each user U_i also verifies received $Sig_i(T_i)$ by calculating H_i and checking that $e(Sig_i(T_i), P) = e(H_i Y_i)$. If the result of the function is equal, then $Sig_i(T_i)$ is verified.

Round 2 - Step 2: Each user does pairing operation to generate, sign, and broadcast the message $Sig_i(X_i)$ and verify all received $Sig_i(X_i)$ by these calculations: $H_i = SHA\_512(X_i)$ and $Sig_i(X_i) = a_i H_i$.

The verification of the received U_i's signatures and integrity checking of X_i are satisfied by calculation of $H_i = SHA-512(X_i)$ and checking the equality of $e(Sig_i(X_i), P) = e(H_i Y_i)$.

Round 2 - Step 3: Each user simultaneously calculates the common key K as described in Step 3 of Section 3.4.1. This step does not require any security contribution.

3.4.3. Observations and efficiency analysis of GKA

In light of both implementation results of the protocol, the total efficiency costs in terms of computation, communication, and round complexities including the whole parties during Round 1 and Round 2 are available in Table 1 and Table 2, respectively.

Table 1. Implementation results of GKA-based steps.

GKA - protocol steps	Computational cost (ms)		Communication cost (bits)
Round 1 - Step 1	4	$\Theta(1)$	$n \times 1024$
Round 2 - Step 2	3	$\Theta(1)$	$n \times 1024$
Round 2 - Step 3	6	$\Theta(1)$	-
Total cost of protocol	13	$\Theta(1)$	$n \times 2048 \rightarrow \Theta(n)$

Table 2. Implementation results of GKA protocol with contributions.

GKA - protocol steps with contributions	Computational cost (ms)		Communication cost (bits)
Round 1 - Step 1	28	$\Theta(1)$	$n \times 2048$
Round 2 - Step 2	$13 + 5(n-1)$	$\Theta(n)$	$n \times 2048$
Round 2 - Step 3	6	$\Theta(1)$	–
Total cost of protocol	$(42 + 5n)$	$\Theta(n)$	$n \times 4096 \rightarrow \Theta(n)$

- As seen from these results (Tables 1 and 2), only the computational cost is affected by security contributions and its computational complexity increases from a constant to a linear class. On the other hand, the communication cost grows twice but it already belongs to the same linear complexity class due to the used signature scheme. Different signature schemes can add different space costs.

3.5. Implementation and efficiency analysis of GKD protocol - VSS

Setup: This is an offline task of the protocol. The numeric values of input and setup parameters are calculated by using the GMP Library. The p is a 1024-bit prime number and g is a 1024-bit primitive root of Z_p^*. The q

is a 512-bit prime number, where $q \mid (p-1)$. For each user U_i and leader we have a (1024-bit) static private key $x_i \in Z_p^*$, and (1024-bit) static public key Y_i where $Y_i = g^{x_i} \, (mod\, p)$ and $Cert_i = (Y_i, ID_i)$, which is exchanged before the protocol starts.

Now the protocol can be started by one of the group participants, called leader L. Here the process of the protocol is presented in two sections as "GKD - protocol steps" (Figure 3) and "GKD - protocol steps with contributions" (Figure 4).

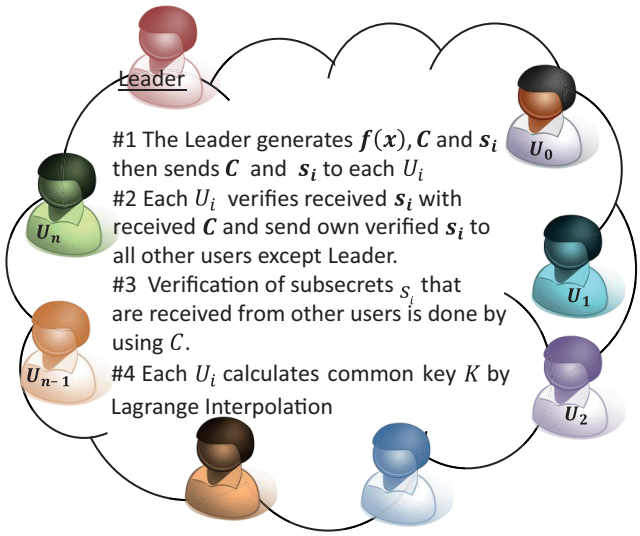

Figure 3. GKD - protocol steps.

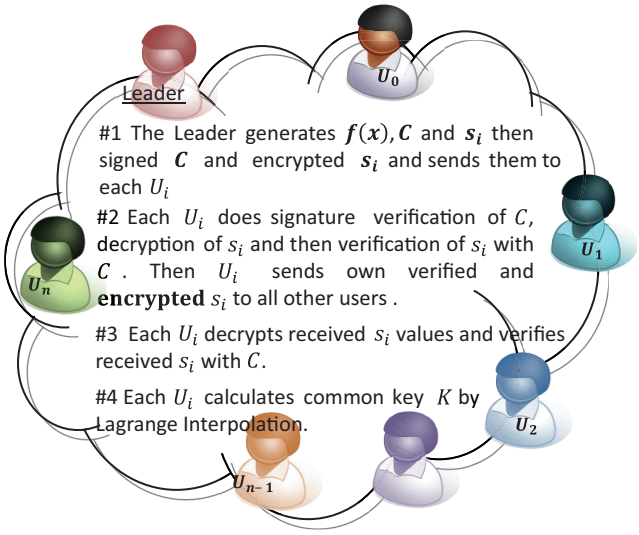

Figure 4. GKD - protocol steps with contributions.

3.5.1. GKD protocol steps as presented in Figure 3

Round 1 - Sharing phase

Leader L announces the public parameters $\langle\rangle param = q, p, g, H$ and the group number n. Here $n = 4 \, and \, t = \lceil (n-1)/2 \rceil = 2$, which defines the second-degree polynomial $f(x) = a_0 + a_1 x + a_2 x^2 \, (mod\, q)$.

In this polynomial, a_0, a_1, $a_2 \in Z_q$ are randomly selected constants of polynomial, and a_0 would like to be shared among the four participants as common secret key s. The commitment vector C is based on DLog commitments [4] as $C = (C_0 = g^{a_0},\ C_1 = g^{a_1},\ C_2 = g^{a_2})\,(mod\,p)$.

Step 1: L generates $f(x)$, C, and su-secrets s_i for each user U_i, using ID_i; $s_i = f(ID_i)\,(mod\,q)$. The leader sends C and s_i to each U_i.

Round 2 - Reconstruction phase

Step 2: Each U_i does verification of subsecret s_i by using received commitment vector C. If s_i is verified properly, it is sent to all other $(n-2)$ users by U_i.

Step 3: The verification of subsecrets s_i that are received from the other $(n-2)$ users is done by using C. This step has to be done by $(n-1)$ users for $(n-2)$ times, and hence the computational complexity is $O(n^2)$ in this step of the protocol.

$g^{s_i}\,(mod\,p) = \prod_{j=0}^{t} C_j^{ij}\,(mod\,p)$ where i, $(i = 1, .., n)$ are indices of the participant and j, $(j = 0, \ldots, t)$ are indices of polynomial coefficients.

Step 4: Lagrange interpolation and common secret key establishment.

Each party U_i has to construct polynomial $f(x)$ from any $(t+1)$ verified subsecrets s_i, which are collected in set W_i and then used in the following Lagrange interpolation formula:

$L_i\,(0) = \lambda_i^w = \prod_{U_i, U_j \in W,\ j \neq i}^{t+1} \frac{ID_j}{ID_j - ID_i}$ and then:

$f\,(0) = \sum_{U_i \in W}^{t+1} L_i(0) s_i$ (note that $s = a_0 = f(0)$) gives the common secret key.

This step is realized $(n-1)\,(t+1)$ times, and then the complexity is here (n^2). The lower bound also is equal to $\Omega(n^2)$ due to $t = \lceil\,(n-1)\,/\,2\,\rceil$, so the complexity value $\theta\left(n^2\right)$ is used in Tables 3 and 4.

Table 3. Implementation results of GKD-based steps.

GKD - protocol steps	Computational cost (ms)		Communication cost (bits)
Round 1 - Step 1	6	$\Theta(1)$	$n \times 1024 + 512$
Round 2-Steps 2	1	$\Theta(1)$	-
Round 2 - Step 3	2	$\Theta(n^2)$	$n^2 \times 512$
Round 2-Step 4	0.04	$\Theta(n^2)$	-
Total cost of protocol	9	$\Theta(n^2)$	$\Theta(n^2)$

Table 4. Implementation results of GKD protocol with contributions.

GKD - protocol steps with contributions	Computational cost (ms)		Communication cost (bits)
Round 1 - Step 1	18	$\Theta(1)$	$(n \times 512) + (t \times 1024)$
Round 2-Steps 2	14	$\Theta(1)$	$n^2 \times 512$
Round 2 - Step 3	6	$\Theta(n^2)$	$n^2 \times 2048$
Round 2-Step 4	0.04	$\Theta(n^2)$	-
Total cost of protocol	38	$\Theta(n^2)$	$\Theta(n^2)$

3.5.2. GKD protocol steps with contributions as depicted in Figure 4

Step 1: The L generates $f(x)$ and C and s_i as aforementioned in Section 3.5.1, Step 1. Before sending C and s_i, L uses the ElGamal signature algorithm in order to sign the commitment vector.

$$H_{Commit} = SHA-512(\,C_0 \parallel C_1 \parallel C_2\,)$$

$$sig_1 = g^r \pmod p \;\; where\, random\, r\, is\, gcd\,(r\,,\,p-1) = 1$$

$$sig_2 = (H_{Commit} - x_L sig_1)\, r^{-1} \;\; mod\,(p-1)$$

L encrypts the calculated s_i by using the receiver's public key Y_i separately via the ElGamal encryption algorithm.

$$Enc^1_{Leader} \;=\; g^{r_{Leader}} \pmod p \;\; where\, random\, r_{Leader}\, is \in \{\,1,\,2,\,..,\,p-1\,\}$$

$$Enc^2_{Leader} \;=\; s_i (Y_i)^{r_{Leader}} \, mod\, p$$

At the end of the Round 1, L sends each encrypted s_i to each related U_i via an open unsecure channel, accompanied with signature pair $(sig_1, sig_2)_C$ of the commitment vector.

Round 2 - Reconstruction phase

Step 2: Each U_i verifies the leader's signature, using the leader's public key Y_L.

$H_{Commit} = SHA\_512\,(\,C_0 \parallel C_1 \parallel C_2\,)$ *and* $g^{H_{commit}} = (Y_L)^{sig_1} sig_1^{sig_2} \;\; mod\, p$

If this equality is satisfied, the receiver U_i is sure about the integrity of the C vector and legitimacy of L. Then each U_i decrypts incoming subsecret $\left(\,Enc^1_{Leader},\, Enc^2_{Leader}\,\right)$, using his own private key x_i.

$$s_i = \; Enc^2_{Leader} \bullet \left(\left(Enc^1_{Leader}\right)^{x_i}\right)^{-1} mod\, p$$

Each U_i uses the receiver's public key Y_j to encrypt his own s_i via the ElGamal encryption algorithm and sends it to the other users of the group to satisfy secrecy and robustness against attacks.

$$Enc^1_{user} \;=\; g^r \, mod\, p \;\; where\, random \;\, r \,\epsilon\, Z^*_p,$$

$$Enc^2_{user} \;=\; s_i (Y_j)^r \, mod\, p \;\; where\, s_i\, is\, subsecret\, of\, U_i$$

Step 3: Each U_i decrypts the received encrypted subsecrets $\left(\,Enc^1_{user},\; Enc^2_{user}\,\right)$ from other users; $s_i = Enc^2_{user}((Enc^1_{user})^{x_j})^{-1} \, mod\, p$. Then these decrypted s_i values are verified by using C; $g^{s_i} \;\, mod\, p \;=\; \prod_{j\,=\,0}^{t} C_j^{i^j}$ $mod\, p$ where $i = 1, .., n$ are the indices of participants and $j = 0, \ldots, t$ are the indices of polynomial coefficients.

Step 4: Each party U_i has to construct polynomial $f(x)$ from any $(\,t+1\,)$ verified subsecrets s_i. Then the common secret key is established by using the Lagrange interpolation formula as explained in Step 4 of Section 3.5.1. Here, this step does not require additional security contribution.

3.5.3. Observations and efficiency analysis of GKD

Our results regarding the protocol implementations are available in Tables 3 and 4.

- The computational overhead is a quadratic value with or without security contributions.

- Most of the computation costs of the protocol steps depend on not only the user number but also on the threshold value t. When user number increases, the size of t will increase and the computation overhead will become very costly. The counts of execution of encryption and decryption algorithms are n^2.
- The proposed protocol is not proper for group key establishment protocol due to very costly communication overhead.
- Finally, round complexity is stable with only two rounds independent of user number.
- The protocol information is theoretically secure even when the adversary has unlimited computing power. Thus, the adversary simply does not have enough information to break the encryption unless he has $(t+1)$ values [12,13].

4. Conclusion

Efficiency is quantified in terms of computation, communication, and round complexity. The experiments in the presented study are performed by using pairing-based cryptography with and without BLS signature scheme for GKA and VSS with and without ElGamal encryption and signature for GKD. The related efficiency analysis and their comparisons are summarized in Table 5, and the following findings can be useful to choose and evaluate a group key establishment solution:

Table 5. Comparison of measured implementation results of protocol models.

Protocol models	Without contributions - Costs		With contributions - Costs	
	Computation	Communication	Computation	Communication
GKA	$\Theta(1)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$
GKD	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$

1. The proposed contribution of signature algorithms, usage of certification mechanisms to identify the group members, and encryption improves the reliability and robustness of the protocol and gives a chance to use it with open unsecure networks. Hence, the protocol has some prevention of passive and active adversaries with negligible overheads.
2. As depicted in Table 5, computational complexity is linear for GKA and quadratic for GKD, which depends on the number of users in the group. The entity count, communication channel limits, and new participant addition or removal scenarios are the critical parameters to decide on the most suitable protocol model of GKE.
   (a) **The GKA protocol:** The key issues for this model can be stated as follows:
   - If any party establishes the wrong secret, he/she will be aware of the failure only when one of the participants encrypts messages with a wrong secret key and the recipient cannot decrypt it. In that case, the recipient sends an error message and the protocol restarts by picking a new ephemeral secret.
   - Static participant number is an important drawback of the protocol. If a new participant would like to join a group, the protocol has to restart to regenerate a new common key. This can be accepted as an advantage or disadvantage according to protocol expectations, because the new user never knows the old secret key.

(b) **The GKD protocol:** The key issues for this model can be stated as follows:

- Consistency of the received messages can be verified via commitments; correctness is guaranteed this way and wrong secret sharing is impossible until the protocol is completed.
- If new party addition or removal occurs simultaneously in the same amount to/from the group during any round of the protocol, there is no any change in the subsecrets of old parties and the common secret s remains the same. However, if the number of entity additions and removals is not balanced, the subsecrets of old parties and the common secret s will change since the protocol starts with new (t, n) parameters.

3. In GKA, the communicating participants have equal contributions, but in GKD the leader is different from others.
4. Certainly, there is a requirement to solve user count limits in groups for GKE protocols because every entity in an open unsecure network should be identified to securely communicate with each other as a closed group.
5. Furthermore, as a future study a formal security analysis of the protocol models with the proposed contributions is necessary to truly verify the approach.

Acknowledgment

We would like to thank Asst Prof Dr Selma Tekir from the Department of Computer Engineering of İzmir Institute of Technology, Turkey, for all her reviewing of the draft versions of this article.

References

[1] Lin CH, Lin HH, Chang JC. Multiparty key agreement for secure teleconferencing. In: SMC 2006 Conference on System, Man, and Cybernetics; 8–11 October 2006; Taipei, Taiwan. New York, NY, USA: IEEE. pp. 3702-3707.

[2] Yoa AC. Protocols for secure computations. In: SFCS 1982 23rd Annual Symposium on Foundations of Computer Sciences; 3–5 November 1982; Chicago, IL, USA. New York, NY, USA: IEEE. pp. 160-164.

[3] Feldman P. A practical scheme for non-interactive verifiable secret sharing. In: SFCS 1987 28th Annual Symposium on Foundations of Computer Science; 12–14 October 1987; Washington, DC, USA. New York, NY, USA: IEEE. pp. 427-438.

[4] Boneh D, Franklin M. Identity-based encryption from the Weil pairing. SIAM J Comput 2003; 32: 586-615.

[5] Chor B, Goldwasser S, Micali S, Awerbuch B. Verifiable secret sharing and achieving simultaneity in the presence of faults. In: SFCS 1985 26th Annual Symposium on Foundations of Computer Science; 21–23 October 1987; Washington, DC, USA. New York, NY, USA: IEEE Computer Society. pp. 383-395.

[6] Aslanoğlu R. Group key establishment protocols: pairing cryptography and verifiable secret sharing scheme. MSc, İzmir Institute of Technology, İzmir, Turkey, 2013.

[7] Boneh D, Lynn B, Shacham H. Short signatures from the Weil pairing. J Cryptol 2004; 4: 297-319.

[8] Barreto PSLM, Lynn B, Scott M. Efficient implementation of pairing-based cryptosystems. J Cryptol 2004; 4: 321-334.

[9] ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE T Inform Theory 1985; 31: 469-472.

[10] Chefranov AG, Mahmoud AY. ElGamal public key cryptosystem and signature scheme in GU(m, p, n). In: SIN 2010 3rd International Conference on Security of Information and Networks; 7–11 September 2010; Taganrog, Russian Federation. New York, NY, USA: ACM. pp. 164-167.

[11] Wei Q, He J, Shao H. A directed signature scheme and its application to group key initial distribution. In: ICIS 2009 2nd International Conference on Interaction Sciences Information Technology, Culture and Human; 24–26 November 2009; Seoul, Korea. New York, NY, USA: ACM. pp. 265-269.

[12] Rabin T. Robust sharing of secrets when the dealer is honest or cheating. J ACM 1994; 41: 1089-1109.

[13] Badanidiyuru A, Patra A, Choudhury A, Srinathan K, Rangan CP. On the trade-off between network connectivity, round complexity, and communication complexity of reliable message transmission. J ACM 2012; 59: 22.