Energy-Efficient ID-based Group Key Agreement Protocols for Wireless Networks

Chik How Tan[1] and Joseph Chee Ming Teo[2]

[1]NISlab
Department of Computer Science
and Media Technology
Gjøvik University College, Norway
chik.tan@hig.no

[2]Information Communication Institute of Singapore
School of Electrical and Electronic Engineering
Nanyang Technological University
Singapore 639798
jose0002@ntu.edu.sg

Abstract

One useful application of wireless networks is for secure group communication, which can be achieved by running a Group Key Agreement (GKA) protocol. One well-known method of providing authentication in GKA protocols is through the use of digital signatures. Traditional certificate-based signature schemes require users to receive and verify digital certificates before verifying the signatures but this process is not required in ID-based signature schemes. In this paper, we present an energy-efficient ID-based authenticated GKA protocol and four energy-efficient ID-based authenticated dynamic protocols, namely Join, Leave, Merge and Partition protocol, to handle dynamic group membership events, which are frequent in wireless networks. We provide complexity and energy cost analysis of our protocols and show that our protocols are more energy-efficient and suitable for wireless networks.

1 Introduction

One useful application of wireless networks is for group communication. However, as the messages in wireless networks are broadcast in plain, they do not provide secure group communication. To provide secure group communication, a group key agreement (GKA) protocol can be used to establish a common group key known only to the users in the group.

The most common way of providing authentication in GKA protocols is through the use of digital signatures. However, this approach usually requires each group member to verify all messages received, which can be large when group size n is large. In certificate-based Public Key Cryptography, before a user can use the public key of the signer to verify the signature, the user has to first obtain and verify the digital certificate issued by a Certifying Authority (CA) to the signer. This results in additional computational cost. ID-based signature schemes do not require the reception and verification of certificates as there are no requirements for public keys in ID-based schemes.

In 1994, Burmester and Desmedt proposed an efficient and secure Burmester-Desmedt (BD) GKA protocol [2] that is suitable for wireless networks. One intuitive way of providing authentication for the BD protocol is to sign and verify all messages sent and received respectively. Although only *two* signatures are required to be generated, each group member will have to verify $n + 2$ messages. The original paper of BD [2] also did not provide protocols to handle dynamic group membership events such as user join, user leave, network merge and network partition, which occur frequently in wireless networks that have dynamic network topology. One intuitive but inefficient method to handle such events as pointed out in [1] and [10] is to re-execute the BD protocol.

In this paper, we present an authenticated GKA protocol that is based on a variant of the Guillou-Quisquater (GQ) signature scheme [5] and the BD protocol [2]. The GQ signature scheme is an efficient ID-based scheme that is not based on pairing, which has high computational cost [14]. We then compare our scheme with other authenticated versions of the BD protocol and the ID-based Saeednia-Safavi-Naini (SSN) GKA protocol [12], which is also based on the BD protocol. The complexity and energy cost analysis are based on the 133MHz "StrongARM" microprocessor and two communication transceivers commonly

1-4244-0054-6/06/$20.00

used in wireless networks. From the energy cost analysis, we show that our scheme is the most efficient. Next, we present four authenticated dynamic protocols, namely Join, Leave, Merge and Partition protocol to handle dynamic group membership events. Similarly, we provide a complexity and energy cost analysis of our dynamic protocols with the BD and show that the energy consumed by nodes running our dynamic protocols are significantly lower.

This paper is organized as follows: Section 2 describes the related work in this area of research while a variant of the GQ signature scheme is presented in Section 3. We present our proposed protocol in Section 4 followed by its complexity and energy analysis in Section 5 and 6 respectively. Section 7 presents our four dynamic protocols and their complexity and energy analysis are given in Section 8. Finally we conclude in Section 9.

2 Related Work

In 1982, Ingemarsson et al. [7] proposed the first GKA protocol known as ING protocol. Following their work, many GKA protocols such as [15, 2, 10, 4, 12] were proposed. One of these protocols, Burmester-Desmedt (BD) [2] protocol, is an efficient and secure GKA protocol, which Katz and Yung [9] recently provided a rigorous security proof in the standard model. Saeednia and Safavi-Naini [12] proposed an ID-based authenticated GKA protocol (SSN) that is based on the BD in 1998. Although this protocol provides authenticated GKA, the number of exponentiations required to be performed by each user is dependent on the group size n. Furthermore, the authors did not specify any dynamic GKA protocols. Currently, most GKA protocols are analyzed based on their complexity. In wireless networks such as MANETs where nodes are usually *low power energy constrained* devices such as PDAs and sensor nodes, it will be more appropriate to analyze the exact energy consumed by each node for different GKA protocols based on the computational and communication costs.

3 GQ ID-based Signature Scheme

Guillou and Quisquater [5] designed an efficient ID-based Signature scheme (GQ) in 1990. We present a variant of the GQ signature scheme as follows:
Setup : The Private Key Generator (PKG) selects two large primes p' and q' and computes $n = p'q'$. Next, the PKG chooses a large number d that is relatively prime to $\Phi(n)$, where $\Phi()$ is Euler's totient function and calculates e such that $gcd(e, d) = 1$. The PKG also selects a one way hash function $H : \{0,1\}^* \rightarrow \{0,1\}^l$, where l is a security parameter. The parameters `params` are (n, e, H) and the master keys are (p', q', d).
Extract : The PKG verifies the given user identity ID and computes the secret key for the identity as $S_{ID} = H(ID)^d \bmod n$. The secret key S_{ID} is then sent securely to user ID.
Sign : Given a private key S_{ID} and a message M, choose $\tau \in_R Z_n^*$ and compute $c = H(\tau^e, M)$ and $s = \tau \cdot S_{ID}^c \bmod n$. Then $\sigma = (s, c)$ is the signature of M.
Verify : The signature $\sigma = (s, c)$ of an identity ID on a message M is valid if the equation $c = H(s^e \cdot (H(ID))^{-c}, m)$ holds good.

4 Proposed ID-based Authenticated Group Key Agreement Protocol

The proposed authenticated GKA protocol is ID-based and uses batch verification based on a variant of the GQ signature scheme. We assume that only honest and trusted nodes are participating in the GKA.

Let $G = \{U_1, \cdots, U_n\}$ be the initial group of n users where $U_i = ID_i$ for $i \in \{1, \cdots, n\}$ refers to the given identity of user U_i. We consider a ring structure among the users of G where the users' indices can be considered on the circulation of $\{1, \cdots, n\}$. The proposed protocol consist of two rounds during which each user U_i will broadcast their key materials z_i and X_i in Round 1 and Round 2 respectively for group key computation. The protocol works as follows:
Setup : The PKG first selects two large (512-bit) prime numbers p' and q' and calculates $n = p'q'$. Next, the PKG chooses a large number d that is relatively prime to $\Phi(n)$ and calculates e such that $gcd(e, d) = 1$. The PKG also selects two random large primes, q (160-bit) and p (1024-bit), such that q divides $p-1$ for the GKA protocol. Next, an element $g \in Z_p^*$ of order q is selected as the generator. Finally, the PKG selects a one way hash function $H : \{0,1\}^* \rightarrow \{0,1\}^l$, where l is a security parameter. The parameters `params` are (n, e, p, q, g, H) and the master keys are (p', q', d).
Extract : The PKG verifies the 32-bit identity $U_i = ID_i$ of user U_i and computes the secret key for U_i as $S_{U_i} = H(U_i)^d \bmod n$. The secret key S_{U_i} is then sent securely to U_i.
Round 1 : Each user U_i for $i \in \{1, \cdots, n\}$ first selects $r_i \in Z_q^*$ and $\tau_i \in Z_n^*$ and computes $z_i = g^{r_i} \bmod p$ and $t_i = \tau_i^e \bmod n$. U_i then broadcast $m_i = U_i||z_i||t_i$ (where $||$ denotes concatenation of messages).
Round 2 : Each user U_i first computes :

$$X_i = \left(\frac{z_{i+1}}{z_{i-1}}\right)^{r_i} = g^{r_i r_{i+1} - r_{i-1} r_i} \bmod p \quad (1)$$

Next, U_i computes $\mathcal{Z} = \prod_{i=1}^{n} z_i \bmod p$, $\mathcal{T} = \prod_{i=1}^{n} t_i \bmod n$ and $c = H(\mathcal{T}, \mathcal{Z})$. U_i then computes $s_i = \tau_i \cdot S_{U_i}^{c} \bmod n$ and produces the signature $\sigma_i = (s_i, c)$. U_i stores $\mathcal{Z}$ and c and broadcasts $m_i' = U_i||X_i||s_i$. It is noted that U_1 is assumed to be a trusted controller such that U_1 will be the last user to broadcast its message m_1' after all the other users $U_j \neq U_1$ have broadcast their messages m_j'.

Authentication and Key Computation: Each U_i verifies the received messages $m_j' \neq m_i'$ by using the stored $\mathcal{Z}$ and c to check the equation:

$$c = H((\prod_{i=1}^{n} s_i)^e \cdot (\prod_{i=1}^{n} H(U_i))^{-c}, \mathcal{Z}) \quad (2)$$

If equation (2) is correct, U_i will proceed to verify the X_i values broadcast in Round 2. However, if equation (2) is incorrect, then all members will retransmit again. From equation (1), we have $X_i = g^{r_i r_{i+1} - r_{i-1} r_i} \bmod p$ for $i \in \{1, \ldots, n\}$, where $r_0 = r_n$ and $r_{n+1} = r_1$. By simple computation, we have the following lemma:

Lemma 1 $\prod_{i=1}^{n} X_i = 1 \bmod p$.

Using *Lemma 1*, we can check whether the X_i sent by group member U_i is genuine. If $\prod_{i=1}^{n} X_i \neq 1 \bmod p$, it means that at least one of the X_i is incorrect. Then, all members will retransmit again. If each U_i correctly verifies that $\prod_{i=1}^{n} X_i = 1 \bmod p$, then U_i computes the common group key K as follows:

$$K = \prod_{i=1}^{n} g^{r_i r_{i+1}} \bmod p = g^{r_1 r_2 + \cdots + r_n r_1} \bmod p \quad (3)$$

where $r_0 = r_n$ and $r_{n+1} = r_1$.

5 Complexity Analysis

Table 1 presents the complexity analysis of different protocols to achieve authenticated BD and the Saeednia-Safavi-Naini (SSN) scheme [12]. The first protocol refers to our proposed GKA scheme in Section 4. The second protocol uses 194-bit ID-based SOK signature scheme [13] to provide authentication for BD. The third and fourth protocols are the BD with 160-bit ECDSA and BD with 1024-bit DSA signature scheme respectively. The last protocol, 1024-bit SSN scheme, uses ID-based cryptography. Although the last protocol do not require any signature generation and verifications, the number of modular exponentiations required is $2n+4$. All other protocols requires only three modular exponentiations for the BD GKA protocol.

Table 1. Complexity Analysis for Authenticated BD GKA

	Our Prop. sch.	BD with SOK	BD with ECDSA	BD with DSA	SSN sch.
Exp.	3	3	3	3	a
Msg Tx	2	2	2	2	2
Msg Rx	b	b	b	b	b
Cert Tx	-	-	1	1	-
Cert Rx	-	-	$n-1$	$n-1$	-
Cert Ver	-	-	$n-1$	$n-1$	-
MapToPt	-	$n-1$	-	-	-
Sign Gen	1	1	1	1	-
Sign Ver	1	$n-1$	$n-1$	$n-1$	-

$a: 2n+4 \qquad b: 2(n-1)$

All protocols, including SSN scheme, require two message transmission and $2(n-1)$ messages to be received by each user. The third and fourth protocol require each user to transmit their certificate in the first message as well as receive and verify $n-1$ certificates from other users. The second protocol is based on pairing and requires $n-1$ MapToPoint operations.

All protocols except the SSN scheme requires one signature generation. This signature generation is done in Round 2 of the GKA for the second, third and fourth protocol where each user signs the message $m_i = U_i||z_i||X_i||\prod_{i=1}^{n} z_i$ to provide authentication for both keying materials z_i and X_i broadcast in Rounds 1 and 2 of the BD respectively. In terms of signature verifications, our proposed protocol is the most efficient.

6 Energy Analysis

In this section, we perform the total energy consumption cost analysis of performing authenticated BD using the 133MHz SA-1110 "StrongARM" microprocessor and two different communication transceivers, namely the 100kbps radio transceiver module and the IEEE 802.11 Spectrum24 WLAN card. We then present a graph that illustrates the total energy consumed by each node while performing authenticated GKA using the protocols shown in Table 1 with the "StrongARM" microprocessor and the two transceiver modules for group size $n = 10, 50, 100$ and 500.

Computational Energy Cost

Table 2 shows the computational energy consumption costs, the computational timing costs of the 133MHz "StrongARM" microprocessor and the computational timing costs of the Pentium III 450MHz

Table 2. Computational Energy Cost

		133MHz StrongARM		450MHz P-III
Mod. Exp.		$9.1mJ$	$37.92ms$	$8.8ms$
MapToPoint		$18.4mJ$	$76.67ms$	$17.78ms$
Tate Pairing		$47.0mJ$	$191.5ms$	$44.4ms$
Scalar Mul.		$8.8mJ$	$36.67ms$	$8.5ms$
Sign. Gen.	DSA	$9.1mJ$	$37.92ms$	$8.8ms$
	ECDSA	$8.8mJ$	$36.67ms$	$8.5ms$
	SOK	$17.6mJ$	$73.33ms$	$17ms$
	GQ	$18.2mJ$	$75.83ms$	$17.6ms$
Sign. Ver.	DSA	$11.1mJ$	$46.33ms$	$10.75ms$
	ECDSA	$10.9mJ$	$45.42ms$	$10.5ms$
	SOK	$137.7mJ$	$573.75ms$	$133.2ms$
	GQ	$18.2mJ$	$75.83ms$	$17.6ms$

Table 3. Communication Energy Cost

	100kbps Transceiver	WLAN Card
Tx per bit	$10.8\mu J$	$0.66\mu J$
Rx per bit	$7.51\mu J$	$0.31\mu J$
Tx. 263-Bytes DSA cert	$22.72mJ$	$1.38mJ$
Rx. 263-Bytes DSA cert	$15.8mJ$	$0.64mJ$
Tx. 86-Bytes ECDSA cert	$7.43mJ$	$0.45mJ$
Rx. 86-Bytes ECDSA cert	$5.17mJ$	$0.21mJ$
Tx. DSA/ECDSA sign.[1]	$3.46mJ$	$0.21mJ$
Rx. DSA/ECDSA sign.[1]	$2.40mJ$	$0.1mJ$
Tx. SOK sign.[2]	$4.19mJ$	$0.26mJ$
Rx. SOK sign.[2]	$2.91mJ$	$0.12mJ$
Tx. GQ sign.[3]	$12.79mJ$	$0.78mJ$
Rx. GQ sign.[3]	$8.89mJ$	$0.36mJ$

[1] DSA/ECDSA signature (r, s), both $r, s = 160$-bits.
[2] SOK signature (S_1, S_2), both $S_1, S_2 = 194$-bits.
[3] GQ signature (s, c), $s = 1024$-bits and $c = 160$-bits.

microprocessor (P3-450MHz) for performing different cryptographic and signature operations.

From [3], we obtained the energy consumption cost of modular exponentiation for the "StrongARM" microprocessor to be $9.1mJ$. As the "StrongARM" microprocessor power consumption is 240mW [3], we can obtain the timing cost of modular exponentiation to be $\frac{9.1mJ}{240mW} = 37.92\ ms$. Based on the MIRACL software library [11], we obtained the computational timing costs of modular exponentiation ($8.8ms$) and other cryptographic operations executed on a Pentium III 450MHz (P3-450MHz) microprocessor in Table 3. From these information, we can extrapolate and estimate the time taken α (ms) for primitive cryptographic operation Y (e.g. Tate Pairing) on the "StrongARM" microprocessor as follows:

$$\alpha = \frac{\gamma\ ms}{8.8\ ms} \times 37.92\ ms \tag{4}$$

where γ refers to the time taken for primitive operation Y on the P3-450MHz microprocessor. Next, we estimate the energy consumed β (mJ) for primitive operation Y on the 133MHz "StrongARM" microprocessor as $\beta = 240mW \times \alpha\ ms$. With these information, we obtained the energy costs in Table 2. The timing cost of the Tate Pairing operation was given to be $20ms$ on the Pentium III 1GHz (P3-1GHz) microprocessor [11]. To obtain the equivalent timing cost on the P3-450MHz microprocessor, we scale down by a factor of $\frac{1000MHz}{450MHz} = 2.22$ to obtain $44.4ms$. From [11], the timing costs of Identity-based Encryption (IBE) Encrypt and Decrypt were given to be $35ms$ and $27ms$ respectively on the P3-1GHz. As the IBE Encrypt requires one additional MapToPoint operation than IBE Decrypt, we were able to obtain the timing cost of MapToPoint operation to be $35 - 27 = 8ms$. To get the timing cost of the MapToPoint operation on the P3-450MHz, we scale down by a factor of 2.22 to get $17.78ms$.

Communications Energy Cost

Table 3 shows the communication energy costs comparison using the 100kbps transceiver module and the IEEE 802.11 Spectrum24 LA-4121 WLAN card. Using the information given in [3] and [6] for the 100kbps transceiver module and [8] for the WLAN card, we obtained the transmission and reception cost per bit of the two transceiver modules. We then obtained the transmission and reception energy costs of transmitting and receiving certificates and signatures.

Energy Consumption Cost Results

By considering the complexity in Table 1 and the energy costs in Tables 2 and 3, we obtained the graph in Figure 1 that shows the total energy consumption costs of each node using the 133MHz "StrongARM" microprocessor with either 100kbps transceiver module or WLAN card. The total energy costs include the transmission and reception costs of all messages as well as the total computational costs in each respective GKA protocol. The figure clearly shows that our proposed scheme is the most energy-efficient when using either the 100kbps transceiver module (i) or WLAN card (j).

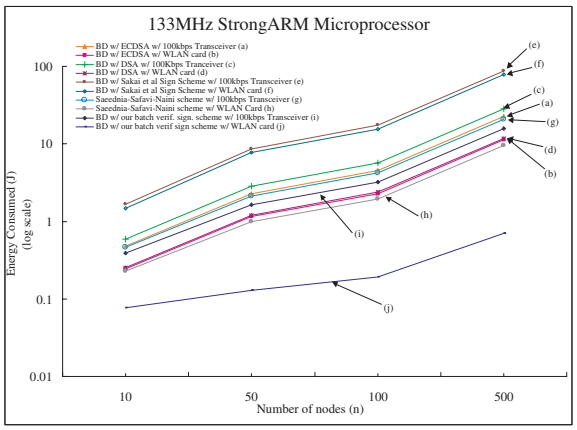

Figure 1. Energy Consumption Costs.

7 Dynamic Protocols

In this section, we present four authenticated dynamic protocols, namely the Join, Leave, Merge and Partition protocol, that can be used to efficiently handle dynamic group membership events. We assume that all group members taking part in the dynamic protocols with possession of the current group key K are trusted nodes. Our proposed dynamic protocols use symmetric key cryptography, which studies [3] and [6] have shown to have energy requirements of orders of magnitude lower than modular exponentiations.

Join Protocol

Let $G = \{U_1, \cdots, U_n\}$ be the current group and U_{n+1} be the new user joining the group. We divide G into two parts, $\{U_1, U_n\}$, which has the two users actively involved in the Join Protocol and $\{U_2, \cdots, U_{n-1}\}$, which consists of the rest of the group. We consider a ring structure among the users of G with U_{n+1} joining G in between U_n and U_1 to form the new group $G' = \{U_1, \cdots, U_{n+1}\}$. The Join Protocol consists of three rounds and works as follows:

Round 1: The new node U_{n+1} first chooses a random $r_{n+1} \bmod q$ and computes $z_{n+1} = g^{r_{n+1}} \bmod p$. Next, U_{n+1} signs $U_{n+1}||z_{n+1}$ using the variant of the GQ signature scheme (Section III) to obtain the signature σ_{n+1} and broadcasts $m_{n+1} = U_{n+1}||z_{n+1}||\sigma_{n+1}$.

Round 2:

(1) U_1 first verifies the signature σ_{n+1}. Next, U_1 chooses a new random $r_1' \in Z_q^*$ and computes K^* as follows:

$$\begin{aligned} K^* &= K \cdot (z_2 \cdot z_n)^{-r_1} \cdot (z_2 \cdot z_{n+1})^{r_1'} \bmod p \\ &= g^{r_1' r_2 + \cdots + r_{n-1} r_n + r_{n+1} r_1'} \bmod p. \end{aligned} \quad (5)$$

U_1 then encrypts $K^*||U_1$ using the current group key K and a symmetric key encryption $E_k(m)$ (where m is the message for encryption and k is the secret key) to obtain $E_K(K^*||U_1)$ and broadcasts $m_1' = U_1||E_K(K^*||U_1)$ to current group G.

(2) U_n verifies the signature σ_{n+1}. U_n then computes the DH key $K_{U_nU_{n+1}} = g^{r_n r_{n+1}} = (z_{n+1})^{r_n} \bmod p$, which it shares with U_{n+1}. Next, U_n encrypts $K_{U_nU_{n+1}}||U_n$ using K to obtain $E_K(K_{U_nU_{n+1}}||U_n)$. U_n then signs $E_K(K_{U_nU_{n+1}}||U_n)||z_n$ using the variant of the GQ signature scheme to obtain the signature σ_n'' and broadcasts $m_n'' = U_n||E_K(K_{U_nU_{n+1}}||U_n)||z_n||\sigma_n''$.

Round 3 :

(1) U_{n+1} first verifies the signature σ_n''. Next, U_{n+1} computes $K_{U_nU_{n+1}} = g^{r_n r_{n+1}} = (z_n)^{r_{n+1}} \bmod p$.

(2) U_n first decrypts $E_K(K^*||U_1)$ in m_1' to obtain K^* and the identity U_1. U_n then checks if the identity U_1 is decrypted correctly to ensure the validity of K^*. Next, U_n encrypts $K^*||U_n$ using $K_{U_nU_{n+1}}$ to obtain $E_{K_{U_nU_{n+1}}}(K^*||U_n)$ and transmits $m_n''' = U_n||E_{K_{U_nU_{n+1}}}(K^*||U_n)$ to U_{n+1}.

Key Computation :

(1) U_{n+1} decrypts $E_{K_{U_nU_{n+1}}}(K^*||U_n)$ in m_n''' to obtain K^* and the identity U_n. U_{n+1} checks if the identity U_n was decrypted correctly to ensure the validity of K^*.

(2) All users $U_i \neq U_1, U_n$ for $i \in \{2, \cdots, n-1\}$ decrypts $E_K(K^*||U_1)$ from m_1' and $E_K(K_{U_nU_{n+1}}||U_n)$ from m_n'' using the current group key K to obtain K^*, the identity U_1, $K_{U_nU_{n+1}}$ and the identity U_n respectively. Next, each U_i checks that the identities U_1 and U_n are decrypted correctly to ensure the validity of K^* and $K_{U_nU_{n+1}}$.

(3) Finally, all users including U_{n+1} compute the new key K' as follows:

$$\begin{aligned} K' &= K^* \cdot K_{U_nU_{n+1}} \bmod p \\ &= g^{r_1' r_2 + \cdots + r_n r_{n+1} + r_{n+1} r_1'} \bmod p \end{aligned} \quad (6)$$

Merge Protocol

A merge occurs when two or more subgroups are combined into a single group. Let $G_A = \{U_1, \cdots, U_n\}$ denote all users in Group A with group key K_A and $G_B = \{U_{n+1}, U_{n+2}, \cdots, U_{n+m}\}$ denote all users in Group B with group key K_B. We divide G_A into two parts, $\{U_1, U_n\}$ and $\{U_2, \cdots, U_{n-1}\}$ and G_B into $\{U_{n+1}, U_{n+m}\}$ and $\{U_{n+2}, \cdots, U_{n+m-1}\}$. We consider a ring structure for G_A, G_B and the merged group $G' = G_A \cup G_B = \{U_1, \cdots, U_{n+m}\}$. The Merge Protocol consists of three rounds and works as follows:

Round 1 :

(1) U_1 first selects a new random $r_1' \in Z_q^*$ and computes $\tilde{z}_1 = g^{r_1'} \bmod p$. Next, U_1 signs $U_1||\tilde{z}_1||z_n$ using the variant of the GQ signature scheme to produce the signature σ_1' and broadcasts $m_1' = U_1||\tilde{z}_1||z_n||\sigma_1'$.

(2) U_{n+1} selects new random $r'_{n+1} \in Z_q^*$, computes $\tilde{z}_{n+1} = g^{r'_{n+1}} \bmod p$ and signs $U_{n+1}||\tilde{z}_{n+1}||z_{n+m}$ using the variant of the GQ signature scheme to obtain the signature σ'_{n+1} and broadcasts $m'_{n+1} = U_{n+1}||\tilde{z}_{n+1}||z_{n+m}||\sigma'_{n+1}$.

Round 2 :

(1) U_1 verifies the signature σ'_{n+1}. Next, U_1 extracts $\tilde{z}_{n+1}$ from m'_{n+1} and computes the DH key $K_{U_1U_{n+1}} = g^{r'_1 r'_{n+1}} = (\tilde{z}_{n+1})^{r'_1} \bmod p$ to be shared with U_{n+1}. U_1 then computes:

$$\begin{aligned} K_A^* &= K_A \cdot (z_2 \cdot z_n)^{-r_1} \cdot (z_2 \cdot z_{n+m})^{r'_1} \\ &= g^{r'_1 r_2 + \cdots + r_{n-1} r_n + r_{n+m} r'_1} \bmod p \end{aligned} \quad (7)$$

Next, U_1 encrypts $K_A^*||U_1$ using Group A's current key K_A and $K_{U_1U_{n+1}}$ to obtain $E_{K_A}(K_A^*||U_1)$ and $E_{K_{U_1U_{n+1}}}(K_A^*||U_1)$ respectively. Finally, U_1 broadcasts the message $m''_1 = U_1||E_{K_A}(K_A^*||U_1)||E_{K_{U_1U_{n+1}}}(K_A^*||U_1)$.

(2) U_{n+1} first verifies the signature σ'_1. Next, U_{n+1} extracts $\tilde{z}_1$ from m'_1 and computes $K_{U_1U_{n+1}} = g^{r'_1 r'_{n+1}} = (\tilde{z}_1)^{r'_{n+1}} \bmod p$, the DH key shared with U_1. U_{n+1} then computes K_B^*:

$$\begin{aligned} K_B^* &= K_B \cdot (z_n \cdot z_{n+2})^{r'_{n+1}} \cdot (z_{n+2} \cdot z_{n+m})^{-r_{n+1}} \\ &= g^{r_n r'_{n+1} + \cdots + r_{n+m-1} r_{n+m}} \bmod p \end{aligned} \quad (8)$$

Next, U_{n+1} encrypts $K_B^*||U_{n+1}$ using Group B's current key K_B and $K_{U_1U_{n+1}}$ to obtain $E_{K_B}(K_B^*||U_{n+1})$ and $E_{K_{U_1U_{n+1}}}(K_B^*||U_{n+1})$ respectively. Finally, U_{n+1} broadcasts $m''_{n+1} = U_{n+1}||E_{K_B}(K_B^*||U_{n+1})||E_{K_{U_1U_{n+1}}}(K_B^*||U_{n+1})$.

Round 3 :

(1) Group B users $U_j \neq U_{n+1}$ for $j \in \{n+2, \cdots, n+m\}$ first decrypts $E_{K_B}(K_B^*||U_{n+1})$ to obtain K_B^* and the identity U_{n+1}. Next, each U_j checks if the identity U_{n+1} was decrypted correctly to ensure that K_B^* is valid and stores K_B^* for key computation later.

(2) U_1 decrypts $E_{K_{U_1U_{n+1}}}(K_B^*||U_{n+1})$ in message m''_{n+1} to obtain K_B^* and the identity U_{n+1}. Next, U_1 checks if the identity U_{n+1} was decrypted correctly to ensure the validity of K_B^*. U_1 then encrypts $K_B^*||U_1$ using Group A's key K_A to obtain $E_{K_A}(K_B^*||U_1)$. Finally, U_1 broadcasts $m'''_1 = U_1||E_{K_A}(K_B^*||U_1)$ to G_A.

(3) U_{n+1} decrypts $E_{K_{U_1U_{n+1}}}(K_A^*||U_1)$ in message m''_1 to obtain K_A^* and the identity U_1. Next, U_{n+1} checks if the identity U_1 was decrypted correctly to ensure that K_A^* is valid. U_{n+1} then encrypts $K_A^*||U_{n+1}$ using Group B's key K_B to obtain $E_{K_B}(K_A^*||U_{n+1})$. Finally, U_{n+1} broadcasts $m'''_{n+1} = U_{n+1}||E_{K_B}(K_A^*||U_{n+1})$ to G_B.

Key Computation :

(1) Group A users $U_i \neq U_1$ for $i \in \{2, \cdots, n\}$ first decrypts $E_{K_A}(K_A^*||U_1)$ in message m''_1 and $E_{K_A}(K_B^*||U_1)$ in m'''_1 to obtain K_A^*, K_B^* and the identity U_1. Each U_i then verifies that the identity U_1 decrypted from both messages m''_1 and m'''_1 is correct to ensure the validity of K_A^* and K_B^* respectively.

(2) Group B users $U_j \neq U_{n+1}$ for $j \in \{n+2, \cdots, n+m\}$ first decrypts $E_{K_B}(K_A^*||U_{n+1})$ in m'''_{n+1} to obtain K_A^* and U_{n+1}. Each U_j then verifies that the identity U_1 decrypted is correct to ensure the received K_A^* is valid.

(3) Finally, all users in the merged group G' compute the new group key K' as follows:

$$\begin{aligned} K' &= K_A^* \cdot K_B^* \bmod p \\ &= g^{r'_1 r_2 + \cdots + r_n r'_{n+1} + \cdots + r_{n+m} r'_1} \bmod p. \end{aligned} \quad (9)$$

Leave Protocol

Let $G = \{U_1, \cdots, U_n\}$ denote the current group and $G' = G \setminus U_l$ denotes the new group. We also consider a ring structure for G and G'. The Leave Protocol comprises of two rounds and works as follows:

Round 1 :All remaining *odd-indexed* users $U_j \neq U_l$ for $j \in \{1, 3, 5, \cdots\}$ select new randoms $r'_j \in Z_q^*$ and $\bar{\tau}_j \in Z_n^*$ and computes $z'_j = g^{r'_j} \bmod p$ and $t'_j = \bar{\tau}_j^e \bmod n$. U_j then broadcasts $m_j = U_j||z'_j||t'_j$.

Round 2 : All users $U_i \neq U_l$ compute X'_i as follows:

$$X'_i = \begin{cases} \left(\frac{z_{i+1}}{z_{i-1}}\right)^{r_i} \text{ if } i \neq l, l-1, l+1, \\ \left(\frac{z_{l+1}}{z_{l-2}}\right)^{r_{l-1}} \text{ if } i = l-1, \\ \left(\frac{z_{l+2}}{z_{l-1}}\right)^{r_{l+1}} \text{ if } i = l+1. \end{cases}$$

Next, each $U_i \in G'$ computes $\bar{\mathcal{Z}} = \prod_{i=1}^n z_i \bmod p$, $\bar{\mathcal{T}} = \prod_{i=1}^n t_i \bmod n$ (where $z_i = z'_j$ and $t_i = t'_j$ for remaining *odd-indexed* users $U_j \in G'$) and $\bar{c} = H(\bar{\mathcal{T}}, \bar{\mathcal{Z}})$. $U_i \neq U_l$ then computes $\bar{s}_i = \bar{\tau}_i \cdot S_{U_i}^{\bar{c}} \bmod n$ and produces the signature $\bar{\sigma}_i = (\bar{s}_i, \bar{c})$. $U_i \neq U_l$ stores $\bar{\mathcal{Z}}$ and $\bar{c}$ and broadcasts $m'_i = U_i||X'_i||\bar{s}_i$. U_1 is assumed to be a trusted controller such that U_1 will be the last user to broadcast its message m'_1 after all the other users have broadcast their messages m'_j.

Authentication and Key Computation: Each $U_i \in G'$ verifies the received messages $m'_j \neq m'_i$ by using the stored $\bar{\mathcal{Z}}$ and $\bar{c}$ to check the equation:

$$\bar{c} = H((\prod_{i=1}^n \bar{s}_i)^e \cdot (\prod_{i=1}^n H(U_i))^{-\bar{c}}, \bar{\mathcal{Z}}) \quad (10)$$

If equation (10) is correct, U_i proceeds to verify the X'_i values using *Lemma 1.* If $\prod_{i=1, i\neq l}^n X'_i \neq 1 \bmod p$, it means that at least one of the X'_i is incorrect. Then, all members will retransmit again. If each $U_i \in G'$ correctly verifies that $\prod_{i=1, i\neq l}^n X_i = 1 \bmod p$, then each

$U_i \in G'$ computes the new group key K' as follows:

$$K' = \prod_{i=1, i\neq l}^{n} g^{r_i r_{i+1}} = g^{r'_1 r_2 + \cdots + r_n r'_1} \bmod p \quad (11)$$

Partition Protocol

A partition can be seen as multiple users leaving the group. Let $G = \{U_1, \cdots, U_n\}$ be the current group, $\mathcal{L}$ be the group of partitioned/leaving users with the total number of partitioned/leaving users as ℓ_d and $G' = G \setminus \mathcal{L}$ be the new group containing the remaining users. The Partition Protocol consists of two rounds and works as follows:

Round 1 :All remaining *odd-indexed* users $U_j \in G'$ for $j \in \{1, 3, 5, \cdots\}$ select new randoms $r'_j \in Z_q^*$ and $\hat{\tau}_j \in Z_n^*$ and computes $z'_j = g^{r'_j} \bmod p$ and $t'_j = \hat{\tau}_j^e \bmod n$. U_j then broadcasts $m_j = U_j || z'_j || t'_j$.

Round 2 : All remaining users $U_i \in G'$ for $i \in \{1, \cdots, n\}$ compute X'_i using the same method as mentioned in Round 2 of the Leave Protocol.

Next, each remaining user $U_i \in G'$ computes $\hat{\mathcal{Z}} = \prod_{i=1}^{n} z_i \bmod p$, $\hat{\mathcal{T}} = \prod_{i=1}^{n} t_i \bmod n$ (where $z_i = z'_j$ and $t_i = t'_j$ for remaining *odd-indexed* users $U_j \in G'$) and $\hat{c} = H(\hat{\mathcal{T}}, \hat{\mathcal{Z}})$. $U_i \in G'$ then computes $\hat{s}_i = \hat{\tau}_i \cdot S_{U_i}^{\hat{c}} \bmod n$ and produces the signature $\hat{\sigma}_i = (\hat{s}_i, \hat{c})$. $U_i \in G'$ stores $\hat{\mathcal{Z}}$ and $\hat{c}$ and broadcasts $m'_i = U_i || X'_i || \hat{s}_i$. U_1 is assumed to be a trusted controller such that U_1 will be the last user to broadcast its message m'_1 after all the other users have broadcast their messages m'_j.

Authentication and Key Computation: Each $U_i \in G'$ verifies the received messages $m'_j \neq m'_i$ by using the stored $\hat{\mathcal{Z}}$ and $\hat{c}$ to check the equation:

$$\hat{c} = H((\prod_{i=1}^{n} \hat{s}_i)^e \cdot (\prod_{i=1}^{n} H(U_i))^{-\hat{c}}, \hat{\mathcal{Z}}) \quad (12)$$

If equation (12) is correct, U_i proceeds to verify the X'_i values using *Lemma 1*. If $\prod_{i=1, i\notin\mathcal{L}}^{n} X'_i \neq 1 \bmod p$, it means that at least one of the X'_i is incorrect. Then, all members will retransmit again. If each $U_i \in G'$ correctly verifies that $\prod_{i=1, i\notin\mathcal{L}}^{n} X_i = 1 \bmod p$, then U_i computes the new group key K' as follows:

$$K' = \prod_{i=1, i\notin\mathcal{L}}^{n} g^{r_i r_{i+1}} = g^{r'_1 r_2 + \cdots + r_n r'_1} \bmod p \quad (13)$$

8 Complexity and Energy Analysis

Complexity Analysis

We compare our dynamic protocols with the authenticated BD protocol using the efficient certificate-based ECDSA signature scheme. As mentioned earlier, the BD protocol is re-executed whenever dynamic group membership events occur as no dynamic protocols have been specified by Burmester and Desmedt in [2]. The complexity of the BD protocol is based on the theoretical evaluation presented in [1] and [10]. The current group size, merging users, merging groups, leaving users, remaining *odd-indexed* users and height of key tree are denoted as n, m, k, ℓ_d, v and h respectively.

Table 4. Complexity Analysis of Dynamic Protocols

Protocol		Comm Cost		Comp Cost		
		Rd	Msgs	Exp.	sign. Gen.	sign. verif.
BD [2]	J	2	$2n+2$	3^a	2	$n+3$
	L	2	$2n-2$	3^a	2	$n+1$
	M	2	$2n+2m$	3^a	2	$n+m$ $+2$
	P	2	$2n-2l_d$	3^a	2	$n-l_d$ $+2$
Prop. Sch.	J	3	5	2^b	1	1
	L	2	$v+n-2$	3^c	1	1
	M	3	$6(k-1)$	4^d	1	1
	P	2	$v+n$ $-2l_d$	3^c	1	1

[a] : All users in BD perform 3 exponentiations.
[b] : *Only* U_1 and U_{n+1} perform 2 exponentiations each.
[c] : *Only* users U_j for j is odd perform 3 exponentiations and the rest performs 2 exponentiations each.
[d] : *Only* U_1 and U_{n+1} perform 4 exponentiations each.

Discussion

Although our Join and Merge Protocols require 1 extra round than the BD Join and Merge Protocol, however both our Join and Merge Protocols require fewer communication messages than the BD Join and Merge Protocols. For exponentiation cost, all users in the BD perform 3 exponentiations each for Join, Leave, Merge and Partition Protocols whereas for our Join Protocol, only U_1 and U_{n+1} performs 2 exponentiations each while the rest need not perform any exponentiations. For our Merge Protocol, only U_1 and U_{n+1} perform 4 exponentiations each while all other users need not perform any exponentiations. Our Leave and Partition Protocols require only users $U_j \notin \mathcal{L}$ for j is odd to perform 3 exponentiations while $U_k \notin \mathcal{L}$ for k is even to perform 2 exponentiations. All our dynamic protocols are also more efficient than the BD in terms of signature generation and verification.

Energy Cost Analysis

In this section, we present the energy cost analy-

sis of our dynamic protocols and compare them with re-executing the authenticated BD protocol using the ECDSA signature scheme. We assume that each node in the group is using the 133MHz "StrongARM" microprocessor and the communication module used is the Spectrum24 WLAN card as presented in Table 2 and 3 respectively. We also assume that the current group size $n = 100$, the number of merging users $m = 20$ and the number of leaving users $\ell_d = 20$. The energy cost results are presented in Table 5.

Table 5. Energy Cost for Dynamic Protocols

		Energy
BD Join	U_1 - U_n	$1.234J$
	U_{n+1}	$2.31J$
Our Join Protocol	U_1	$0.039J$
	U_n	$0.049J$
	U_{n+1}	$0.057J$
	Others	$1.34mJ$
BD Leave	Remain. Users	$1.179J$
Our Leave Protocol	U_j, $j = odd$	$0.160J$
	U_k, $k = even$	$0.150J$
BD Merge	Group A Users	$1.660J$
	Group B Users	$2.532J$
Our Merge Protocol	U_1	$0.079J$
	U_{n+1}	$0.079J$
	Others	$0.986mJ$
BD Partition	Remain. Users	$0.942J$
Our Partition Protocol	U_j, $j = odd$	$0.142J$
	U_k, $k = even$	$0.132J$

Discussion

Based on the energy cost results shown in Table 5, it can be easily seen that our protocols are much more suitable to be implemented in wireless networks where dynamic group membership events occurs frequently. If the BD protocol were to be implemented, then the battery life of each node will be depleted more rapidly due to the high energy costs. On the other hand, our dynamic protocols consume much lower energy as shown in Table 5.

9 Conclusion

We present an energy-efficient initial GKA protocol and four authenticated dynamic GKA protocols that take advantage of the ID-based signature scheme. The complexity and energy cost analysis of our five protocols show that they are much more suitable for wireless networks. The security of our protocols is based on the security of the BD protocol and the GQ signature scheme, both of which have been proven to be secure. Due to the page limit, the detail security analysis of our protocols will be provided in the full version.

References

[1] Y. Amir, Y. Kim, C. Nita-Rotaru, and G. Tsudik. On the performance of group key agreement protocols. *John Hopkins Univ., Center of Networking and Distributed Systems, TR CNDS-2001-5*, 2001.

[2] M. Burmester and Y. Desmedt. A secure and efficient conference key distribution system. In *Advances in Cryptography - Eurocrypt '94*, volume 950 of *LNCS*, pages 275–286, 1995.

[3] D. W. Carman, P. S. Kruss, and B. J. Matt. Constraints and approaches for distributed sensor network security. *NAI Labs TR #00-010*, Sept. 2000.

[4] S. Cho, J. Nam, S. Kim, and D. Won. An efficient dynamic group key agreement for low-power mobile devices. In *ICCSA 2005*, volume 3480 of *LNCS*, pages 498 – 507, Apr. 2005.

[5] L. C. Guillou and J. J. Quisquater. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In *Advances in Cryptology - Crypto '88*, volume 0403 of *LNCS*, pages 216–231, 1990.

[6] A. Hodjat and I. Verbauwhede. The energy cost of secrets in ad-hoc networks. In *IEEE CAS Workshop on Wireless Communication and Networking*, Sept. 2002.

[7] I. Ingemarsson, D. T. Tang, and C. K. Wong. A conference key distribution system. *IEEE Transaction on Information Theory*, 28(5):714 – 720, Sept. 1982.

[8] R. Karri and P. Mishra. Optimizing the energy consumed by secure wireless sessions: wireless transport layer security case study. *Mobile Networks and Applications*, 8(2):177 – 185, Apr. 2003.

[9] J. Katz and M. Yung. Scalable protocols for authenticated group key exchange. In *Advances in Cryptography - Crypto'03*, volume 2729 of *LNCS*, pages 110 – 125, 2003.

[10] Y. Kim, A. Perrig, and G. Tsudik. Tree-based group key agreement. *ACM Transaction on Information and System Security*, 7(1):60 – 96, Feb. 2004.

[11] MIRACL. Multiprecision integer and rational arithmetic c/c++ library. *http://indigo.ie/ mscott/*.

[12] S. Saeednia and R. Safavi-Naini. Efficient identity-based conference key distribution protocols. In *Information Security and Privacy - 3rd Australasian Conference*, volume 1438 of *LNCS*, pages 320–331, 1998.

[13] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, 2000.

[14] N. Saxena, G. Tsudik, and J. H. Yi. Identity-based access control for ad hoc groups. In *ICISC 2004*, volume 3506 of *LNCS*, pages 362–379, 2005.

[15] M. Steiner, G. Tsudik, and M. Waidner. Key agreement in dynamic peer groups. *IEEE Transactions on Parallel and Distributed Systems*, 11(8):769 – 780, Aug. 2000.