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

Chik How Tan¹ and Joseph Chee Ming Teo²

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

²Information 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 used in wireless networks.
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 Φ(n), where Φ() 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}∗ → {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 $S_{ID} = H(ID)^d \mod n$. The secret key $S_{ID}$ is then sent securely to user ID.

Sign : Given a private key $S_{ID}$ and a message M, choose τ ∈ R $Z^*_n$ and compute c = H(τ, M) and s = τ · $S_{ID} \mod n$. Then σ = (s, c) is the signature of M.

Verify : The signature σ = (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 = {U1, · · · , Un} 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, · · · , 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 Φ(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 ∈ $Z^*_n$ 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 $S_{U_i} = H(U_i)^d \mod n$. 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 $t_i \in Z^*_n$ and computes $z_i = g^{r_i} \mod p$ and $t_i = \tau^e \mod n$. $U_i$ then broadcast $m_i = U_i||z_i||t_i$ (where || denotes concatenation of messages).

Round 2 : Each user $U_i$ first computes

$$X_i = \left(\frac{z_{i+1}}{z_{i-1}}\right)^{r_i} = g^{r_i \cdot (r_1, \cdots , r_{i-1})} \mod p \quad (1)$$
Next, $U_i$ computes $Z = \prod_{i=1}^{n} z_i \mod p$. $T = \prod_{i=1}^{n} t_i \mod n$ and $c = H(T, Z)$. $U_i$ then computes $s_i = t_i \cdot S_i \mod n$ and produces the signature $\sigma_i = (s_i, c)$. $U_i$ stores $Z$ and $c$ and broadcasts $m'_i = U_i||X_i||s_i$. It is noted that $U_i$ 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 $Z$ and $c$ to check the equation:

\[
c = H(\prod_{i=1}^{n} s_i)^c \cdot (\prod_{i=1}^{n} H(U_i))^{-c} \mod Z
\]  

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 $X_i = g^{r_i t_{i+1} - r_i t_i} \mod p$ 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** \(\prod_{i=1}^{n} X_i = 1 \mod p\).

Using **Lemma 1**, we can check whether the $X_i$ sent by group member $U_i$ is genuine. If $\prod_{i=1}^{n} X_i \neq 1 \mod p$, 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 $\prod_{i=1}^{n} X_i = 1 \mod p$, then $U_i$ computes the common group key $K$ as follows:

\[
K = \prod_{i=1}^{n} g^{r_i t_{i+1}} \mod p = g^{r_1 t_2 + \cdots + r_n t_1} \mod p
\]

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                        | \(b\)  | \(b\)  | \(b\)  | \(b\)  | 2      |
| 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                      | \(n-1\) | \(n-1\) | \(n-1\) | -      | -      |

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

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

<table>
<thead>
<tr>
<th>Operation</th>
<th>133MHz StrongARM</th>
<th>450MHz P-III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mod. Exp.</td>
<td>9.1mJ 37.92ms 8.8ms</td>
<td>18.4mJ 76.67ms 17.75ms</td>
</tr>
<tr>
<td>MapToPoint</td>
<td>47.0mJ 191.5ms 44.4ms</td>
<td>36.67ms 8.5ms</td>
</tr>
<tr>
<td>Tate Pairing</td>
<td>17.6mJ 73.33ms 17ms</td>
<td>17.6ms 8.5ms</td>
</tr>
<tr>
<td>Scalar Mul.</td>
<td>18.2mJ 75.83ms 17.6ms</td>
<td>8.5ms</td>
</tr>
<tr>
<td>Sign. Gen.</td>
<td>11.1mJ 46.33ms 10.75ms</td>
<td>10.9mJ 54.22ms 10.5ms</td>
</tr>
<tr>
<td>ECDSA</td>
<td>137.7mJ 573.75ms 133.2ms</td>
<td>18.2mJ 75.83ms 17.6ms</td>
</tr>
<tr>
<td>SOK</td>
<td>137.7mJ 573.75ms 133.2ms</td>
<td>18.2mJ 75.83ms 17.6ms</td>
</tr>
<tr>
<td>GQ</td>
<td>137.7mJ 573.75ms 133.2ms</td>
<td>18.2mJ 75.83ms 17.6ms</td>
</tr>
</tbody>
</table>

Table 3. Communication Energy Cost

<table>
<thead>
<tr>
<th>Operation</th>
<th>100kbps Transceiver</th>
<th>WLAN Card</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tx per bit</td>
<td>10.8µJ 0.66µJ</td>
<td>7.51µJ 0.31µJ</td>
</tr>
<tr>
<td>Rx per bit</td>
<td>22.72mJ 1.58mJ</td>
<td>15.8mJ 0.64mJ</td>
</tr>
<tr>
<td>Tx. 263-Bytes DSA cert</td>
<td>7.43mJ 0.45mJ</td>
<td>5.17mJ 0.21mJ</td>
</tr>
<tr>
<td>Rx. 263-Bytes DSA cert</td>
<td>5.17mJ 0.21mJ</td>
<td>3.46mJ 0.21mJ</td>
</tr>
<tr>
<td>Tx. DSA/ECDSA cert</td>
<td>2.40mJ 0.1mJ</td>
<td>2.91mJ 0.12mJ</td>
</tr>
<tr>
<td>Rx. DSA/ECDSA cert</td>
<td>2.40mJ 0.1mJ</td>
<td>12.79mJ 0.78mJ</td>
</tr>
<tr>
<td>Tx. GQ sign</td>
<td>8.89mJ 0.36mJ</td>
<td>3.46mJ 0.21mJ</td>
</tr>
<tr>
<td>Rx. GQ sign</td>
<td>8.89mJ 0.36mJ</td>
<td>2.91mJ 0.12mJ</td>
</tr>
</tbody>
</table>

1 DSA/ECDSA signature (r, s), both r, s = 160-bits.
2 SOK signature (S1, S2), both S1, S2 = 194-bits.
3 GQ signature (s, c), s = 1024-bits and c = 160-bits.

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 

\[
\frac{9.1mJ}{240mW} = 37.92 \text{ ms}
\]

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 \( \alpha \) for primitive cryptographic operation Y (e.g. Tate Pairing) on the “StrongARM” microprocessor as follows:

\[
\alpha = \frac{\gamma \text{ ms}}{8.8 \text{ ms}} \times 37.92 \text{ ms}
\]

where \( \gamma \) refers to the time taken for primitive operation Y on the P3-450MHz microprocessor. Next, we estimate the energy consumed \( \beta \) (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).
protocols with possession of the current group key to handle dynamic group membership events. We assume Partition protocol, that can be used to efficiently handle dynamic protocols, namely the Join, Leave, Merge and Join Protocol have shown to have energy requirements of orders of magnitude lower than modular exponentiations. We consider a ring structure among the users of a group and denote all users in \( G \) by \( U \). Let \( G = \{U_1, \ldots, U_n\} \) 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 \( \{U_2, \ldots, U_{n-1}\} \), 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 \( G' = \{U_1, \ldots, U_{n+1}\} \). The Join Protocol consists of three rounds and works as follows:

**Round 1:** The new node \( U_{n+1} \) first chooses a random \( r_{n+1} \) \( \mod q \) and computes \( z_{n+1} = g^{r_{n+1}} \mod p \). 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 \( \sigma_{n+1} \) and broadcasts \( m_{n+1} = U_{n+1} || z_{n+1} || \sigma_{n+1} \).

**Round 2:**
1. \( U_1 \) first verifies the signature \( \sigma_{n+1} \). Next, \( U_1 \) chooses a new random \( r_1' \in \mathbb{Z}_q^* \) and computes \( K^* \) as follows:

\[
K^* = K \cdot (z_2 \cdot z_n)^{-r_1} \cdot (z_2 \cdot z_{n+1})^{r_1'} \mod p = g^{r_1' r_2 + \cdots + r_n + r_{n+1} + r_1} \mod p. \tag{5}
\]

\( U_1 \) 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 \( \sigma_{n+1} \). \( U_n \) then computes the DH key \( K_{U_n U_{n+1}} = g^{r_{n+1}} = (z_{n+1})^n \mod p \), which it shares with \( U_{n+1} \). Next, \( U_n \) encrypts \( K_{U_n U_{n+1}} || U_n \) using \( K \) to obtain \( E_k(K_{U_n U_{n+1}} || U_n) \). \( U_n \) then signs \( E_k(K_{U_n U_{n+1}} || U_n) || z_n \) using the variant of the GQ signature scheme to obtain the signature \( \sigma_n' \) and broadcasts \( m_n' = U_n || E_k(K_{U_n U_{n+1}} || U_n) || z_n || \sigma_n' \).

**Round 3:**
1. \( U_{n+1} \) first verifies the signature \( \sigma_{n+1} \). Next, \( U_{n+1} \) computes \( K_{U_n U_{n+1}} = g^{r_{n+1}} = (z_n)^{r_{n+1}} \mod p \).
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 that the identity \( U_1 \) was decrypted correctly to ensure the validity of \( K^* \). Next, \( U_n \) encrypts \( K^* || U_n \) using \( K_{U_n U_{n+1}} \) to obtain \( E_k(K_{U_n U_{n+1}} || K^* || U_n) \) and transmits \( m_n'' = U_n || E_k(K_{U_n U_{n+1}} || K^* || U_n) \) to \( U_{n+1} \).

**Key Computation:**
1. \( U_{n+1} \) decrypts \( E_k(K_{U_n U_{n+1}} || K^* || U_n) \) in \( m_n'' \) to obtain \( K^* \) and the identity \( U_n \). \( U_{n+1} \) checks that the identity \( U_n \) was decrypted correctly to ensure the validity of \( K^* \).
2. All users \( U_i \neq U_1, U_n \) for \( i \in \{2, \ldots, n-1\} \) decrypts \( E_k(K^* || U_1) \) from \( m'_1 \) and \( E_k(K_{U_n U_{n+1}} || U_n) \) from \( m_n'' \) using the current group key \( K \) to obtain \( K^* \), the identity \( U_1 \), \( K_{U_n U_{n+1}} \), 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 \( K_{U_n U_{n+1}} \).
3. Finally, all users including \( U_{n+1} \) compute the new key \( K' \) as follows:

\[
K' = K^* \cdot K_{U_n U_{n+1}} \mod p = g^{r_i' r_2 + \cdots + r_{n+1} + r_n + r_i} \mod p \tag{6}
\]

**Join Protocol**

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

**Round 1:**
1. \( U_1 \) first selects a new random \( r_1' \in \mathbb{Z}_q^* \) and computes \( z_1 = g^{r_1'} \mod p \). Next, \( U_1 \) signs \( U_1 || z_1 || z_n \) using the variant of the GQ signature scheme to produce the signature \( \sigma_1' \) and broadcasts \( m'_1 = U_1 || z_1 || z_n || \sigma_1' \).
Next, $U_{n+1}$ selects new random $r'_{n+1} \in Z_q^*$, computes $\tilde{z}_{n+1} = g^{r'_{n+1}} \mod p$ and signs $U_{n+1}||\tilde{z}_{n+1}||z_{n+1}$ using the variant of the GQ signature scheme to obtain the signature $\sigma'_{n+1}$ and broadcasts $m'_{n+1} = U_{n+1}||\tilde{z}_{n+1}||z_{n+1}||\sigma'_{n+1}$.

**Round 2 :**

(1) $U_1$ verifies the signature $\sigma'_{n+1}$. Next, $U_1$ extracts $\tilde{z}_{n+1}$ from $m'_{n+1}$ and computes the DH key $K_{U_1U_{n+1}} = g^{r'_{n+1}} = (\tilde{z}_{n+1})^{r_1} \mod p$ to be shared with $U_{n+1}$. $U_1$ then computes:

\[
K^*_A = K_A \cdot (z_2 \cdot z_{n+2})^{-r_1} \cdot (z_2 \cdot z_{n+2})^{r_1} = g^{r_2r_3+\cdots r_n+r_{n+m}r_1} \mod p \tag{7}
\]

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 $\sigma'_{n+1}$. Next, $U_{n+1}$ extracts $\tilde{z}_1$ from $m'_1$ and computes $K_{U_1U_{n+1}} = g^{r'_{n+1}} = (\tilde{z}_1)^{r_{n+1}} \mod p$, the DH key shared with $U_1$. $U_{n+1}$ then computes $K^*_B$:

\[
K^*_B = K_B \cdot (z_n \cdot z_{n+2})^{r'_{n+1}} \cdot (z_n \cdot z_{n+2})^{-r_{n+1}} = g^{r_2r_3+\cdots r_n+r_{n+m}r_{n+1}} \mod p \tag{8}
\]

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 $K_A||U_1$ using Group A’s key $K_A$ to obtain $E_{K_A}(K_A||U_1)$. Finally, $U_1$ broadcasts $m'''_{n+1} = U_{n+1}||E_{K_A}(K_A||U_1)$ to $G_A$.

(3) $U_{n+1}$ 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, $U_{n+1}$ checks if the identity $U_{n+1}$ was decrypted correctly to ensure that $K^*_B$ 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:
$U_i \in G'$ computes the new group key $K'$ as follows:

$$K' = \prod_{i = 1, i \neq l}^{n} g^{r_i r_{i+1}} = g^{r_1 r_2 + \cdots + r_n r'_1} \mod p \tag{11}$$

**Partition Protocol**

A partition can be seen as multiple users leaving the group. Let $G = \{U_1, \ldots, U_n\}$ be the current group, $L$ be the group of partitioned/leaving users with the total number of partitioned/leaving users as $\ell_d$ and $G' = G \setminus L$ 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 $j \in \{1, 3, 5, \ldots \}$ select new randoms $r'_j \in Z_q^*$ and $\tilde{\tau}_j \in Z_n^*$ and computes $z'_j = g^{r'_j} \mod p$ and $t'_j = \tilde{\tau}_j^e \mod n$. $U_j$ then broadcasts $m_j = U_j || z'_j || t'_j$.

**Round 2** : All remaining users $U_i \in G'$ for $i \in \{1, \ldots, n\}$ 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 $\hat{Z} = \prod_{i=1}^n z_i \mod p$, $\hat{T} = \prod_{i=1}^n t_i \mod n$ (where $z_i = z'_j$ and $t_i = t'_j$ for remaining odd-indexed users $U_j \in G'$) and $\hat{c} = H(\hat{T}, \hat{Z})$. $U_i \in G'$ then computes $\hat{s}_i = \tilde{\tau}_i \cdot S^{t_i}_{U_i} \mod n$ and produces the signature $\hat{\sigma}_i = (\hat{s}_i, \hat{c})$.

$U_i \in G'$ stores $\hat{Z}$ and $\hat{c}$ and broadcasts $m'_i = U_i || X'_i || \hat{\sigma}_i$. $U_j$ is assumed to be a trusted controller such that $U_j$ will be the last user to broadcast its message $m'_i$ after all other users have broadcast their messages $m'_i$.

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

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

If equation (12) is correct, $U_i$ proceeds to verify the $X'_i$ values using Lemma 1. If $\prod_{i=1, i \notin L}^{n} X'_i \neq 1 \mod p$, 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 $\prod_{i=1, i \notin L}^{n} X'_i = 1 \mod p$, then $U_i$ computes the new group key $K'$ as follows:

$$K' = \prod_{i=1, i \notin L}^{n} g^{r_i r_{i+1}} = g^{r_1 r_2 + \cdots + r_n r'_1} \mod p \tag{13}$$

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 $n, m, k, \ell_d, v$ and $h$ respectively.

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Comm Cost</th>
<th>Comp Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>BD [2]</td>
<td>J: 2</td>
<td>2n + 2</td>
</tr>
<tr>
<td></td>
<td>L: 2</td>
<td>2n - 2</td>
</tr>
<tr>
<td></td>
<td>M: 2</td>
<td>2n + 2m</td>
</tr>
<tr>
<td></td>
<td>P: 2</td>
<td>2n - 2l_d</td>
</tr>
</tbody>
</table>

Table 4. Complexity Analysis of Dynamic Protocols

$^a$: 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 $U_j \notin L$ for $j$ is odd to perform 3 exponentiations while $U_k \notin L$ 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>
<thead>
<tr>
<th>Table 5. Energy Cost for Dynamic Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>BD Join</td>
</tr>
<tr>
<td>Our Join</td>
</tr>
<tr>
<td>Protocol</td>
</tr>
<tr>
<td>BD Leave</td>
</tr>
<tr>
<td>Our Leave</td>
</tr>
<tr>
<td>Partition Protocol</td>
</tr>
<tr>
<td>BD Merge</td>
</tr>
<tr>
<td>Merge</td>
</tr>
<tr>
<td>Our Merge</td>
</tr>
<tr>
<td>Protocol</td>
</tr>
<tr>
<td>BD Partition</td>
</tr>
<tr>
<td>Partition Protocol</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

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 occur 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