A Comparison of GKE Protocols based on SIDH

Hiroki Okada[1], Shinsaku Kiyomoto[1] and Carlos Cid[2,3]
[1]*KDDI Research, Inc., Saitama, Japan*
[2]*Royal Holloway, University of London, Egham, U.K.*
[3]*Simula UiB, Norway*

Keywords: Group Key Establishment, Post-quantum Cryptography, Isogeny-based Cryptography, SIDH.

Abstract: End-to-end encryption enables secure communication without releasing the contents of messages to the system server. This is a crucial security technology, in particular to cloud services. *Group Key Establishment* (GKE) protocols are often needed to implement efficient group end-to-end encryption systems. Perhaps the most famous GKE protocol is the *Broadcast Protocol*, proposed by Burmester and Desmedt. In addition, they also proposed the *Star-based Protocol*, *Tree-based Protocol*, and *Cyclic-based Protocol*. These protocols are based on the Diffie-Hellman key exchange protocol, and therefor are not secure against attacks based on quantum computers. Recently, Furukawa *et al.* proposed an efficient GKE protocol by modifying the original Broadcast Protocol into a post-quantum GKE protocol based on the *Supersingular Isogeny Diffie-Hellman key exchange* (SIDH). In this paper, we extend their work by considering the remaining DH-based GKE protocols by Burmester and Desmedt post-quantum versions based on SIDH, and compare their efficiency. As a result, we confirm that the Broadcast Protocol is indeed the most efficient protocol in this post-quantum setting, in terms of both communication rounds and computation time.

1 INTRODUCTION

Secure and efficient key management is long-standing active area of research in cryptography. In many cases, we have a trustworthy central server in the system, which is usually responsible for the efficient key management. However, in the case of cloud services, servers are not necessarily trustworthy. For example, when using chat applications, one usually does not want to let service providers see the contents of chats – we do not want to give the server access to the secret keys, thus end-to-end encryption is usually desirable in this case. When considering end-to-end encrypted group communication, key management is more complicated. Group Key Establishment (GKE) protocols enable group members to efficiently share the group key without giving access to the server.

Perhaps the most famous GKE protocol is *Broadcast Protocol* proposed by Burmester and Desmedt (Burmester and Desmedt, 1994). The authors also proposed the *Star-based Protocol*, *Tree-based Protocol*, and *Cyclic-based Protocol*. All these protocols area based on the Diffie-Hellman key exchange, and are therefore insecure against attacks based on quantum computers.

Recently, Furukawa *et al.* proposed an efficient post-quantum GKE protocol (Furukawa et al., 2018) by modifying the Broadcast Protocol of (Burmester and Desmedt, 1994) to use the *Supersingular Isogeny Diffie-Hellman key exchange* (SIDH), which is believed to be post-quantum secure. The concept of the isogeny-based cryptography was first proposed in (Silverman, 1986). The first concrete isogeny-based Diffie-Hellman type key exchange protocol was proposed in (Rostovtsev and Stolbunov, 2006). The protocol was defined over ordinary elliptic curves, and was originally believed to be post-quantum secure. However, the authors of (Childs et al., 2014) described a subexponential quantum algorithm to compute isogenies between ordinary curves, thus showing that Rostovtsev *et al.*'s protocol is not post-quantum secure. Jao *et al.* (Jao and De Feo, 2011) proposed an isogeny-based Diffie-Hellman type key exchange protocol defined over supersingular elliptic curves, called SIDH, which is believed to post-quantum secure. They are also among the authors of the post-quantum key establishment mechanism based on SIDH that was submitted to NIST's Post-Quantum Cryptography competition (National Institute of Standards and Technology, 2020).

Okada, H., Kiyomoto, S. and Cid, C.
A Comparison of GKE Protocols based on SIDH.
DOI: 10.5220/0010547305070514
In *Proceedings of the 18th International Conference on Security and Cryptography (SECRYPT 2021)*, pages 507-514
ISBN: 978-989-758-524-1

One important use case of GKE protocols are group chat applications. The problem of privacy in these applications has been receiving growing attention from the cryptographic community. It was sometimes suspected that providers of these services might be able to gather the information of chats if they do not support the End-to-End encryption. As a result, in January 2018, Skype started testing new "private conversations" with end-to-end encryption (Microsoft Community, 2018), using the industry standard *Signal Protocol* by Open Whisper Systems (Open Whisper Systems, 2018). Facebook Messenger and Allo (by Google) also use this *Signal Protocol* for End-to-End encryption.

The *Signal Protocol* is a non-federated cryptographic protocol, initially with no academic security analysis publicly provided. Cohn-Gordon *et al.* produced a first academic analysis of the protocol, followed by several other works, e.g. in (Bellare et al., 2017; Herzberg and Leibowitz, 2016; Cohn-Gordon et al., 2018; Rösler et al., 2018; Tanada et al., 2016; Cohn-Gordon et al., 2016). Cohn-Gordon *et al.* reported that there were no major flaws in the design of *Signal Protocol*. We note however that their first study was focused on Signal's two-party key exchange protocol, and did not include group messaging properties (since the implementation of group messaging of the *Signal Protocol* is not specified at the protocol layer (Perrin and Marlinspike, 2016)).

Moreover, the authors reported in (Cohn-Gordon et al., 2018) that "While these users' two-party communications now enjoy very strong security guarantees, it turns out that many of these apps provide, without notifying the users, a weaker property for group messaging: an adversary who compromises a single group member can intercept communications indefinitely.". As a result a Tree based GKE has been proposed for adoption in the *Signal protocols*. Also note that IETF MLS WG (Barnes et al., 2020) tries to generalize the *Signal protocols* to the group setting.

2 PRELIMINARY

2.1 Notation and Definition

- $a \in_R A$: a is selected from the set A uniformly and independently at random.
- Group Key Agreement (GKA): all users participate to key generation; group key constructed based on all user's secrets.
- Group Key Transfer (GKT): a privileged user (group manager / KGC*) selects a key and securely distributes it to the other users.
- Group Key Establishment (GKE): general term including GKA and GKT.

2.2 SIDH Protocol

We briefly explain the SIDH protocol; for full details see (Jao and De Feo, 2011). The fixed public parameters of this scheme are $\{l_A, l_B,\ e_A, e_B, f,\ p = l_A^{e_A} l_B^{e_B} f - 1,\ E_0(\mathbb{F}_{p^2}),\ \{P_A, Q_A\},\ \{P_B, Q_B\}\}$, where l_A and l_B are small primes, e_A, e_B, f so that p is a prime. E_0 is a base supersingular elliptic curve, and $\{P_A, Q_A\}$ and $\{P_B, Q_B\}$ are bases of $E_0[l_A^{e_A}]$ over $\mathbb{Z}/l_A^{e_A}\mathbb{Z}$ and $E_0[l_B^{e_B}]$ over $\mathbb{Z}/l_B^{e_B}\mathbb{Z}$, respectively. The key exchange protocol is summarised as follows.

Alice chooses two random secret elements $m_A, n_A \in_R \mathbb{Z}/l_A^{e_A}\mathbb{Z}$, where m_A, n_A are not divisible by l_A, and computes an isogeny $\phi_A : E_0 \to E_A$ whose kernel is $\langle [m_A]P_A + [n_A]Q_A \rangle$. Alice also computes the image $\{\phi_A(P_B), \phi_A(Q_B)\} \subset E_A$ of the basis $\{P_B, Q_B\}$ using her isogeny ϕ_A. She sends $\{\phi_A(P_B), \phi_A(Q_B)\}$ and E_A to Bob.

Similarly, Bob chooses two random secret elements $\mathbb{Z}/l_B^{e_B}\mathbb{Z}$, where m_B, n_B are not divisible by l_B, and computes an isogeny $\phi_B : E_0 \to E_B$ whose kernel is $\langle [m_B]P_B + [n_B]Q_B \rangle$. Bob then computes $\{\phi_B(P_A), \phi_B(Q_A)\} \subset E_B$ and sends $\{\phi_B(P_A), \phi_B(Q_A)\}$ and E_B to Alice. With this information sent by Bob, Alice computes an isogeny $\phi'_A : E_B \to E_{AB}$ whose kernel is $\{[m_A]\phi_B(P_A), [n_A]\phi_B(Q_A)\}$. Similarly, Bob computes $\phi'_B : E_A \to E_{AB}$ whose kernel is $\{[m_B]\phi_A(P_B),\ [n_B]\phi_A(Q_B)\}$. Alice and Bob can then use the common j-invariant of

$$\begin{aligned} E_{AB} &= \phi'_B(\phi_A(E_0)) = \phi'_A(\phi_B(E_0)) \\ &= E_0/\{[m_A]P_A + [n_A]Q_A, [m_B]P_B + [n_B]Q_B\}, \end{aligned}$$

to generate a secret shared key.

3 GROUP KEY ESTABLISHMENT PROTOCOLS BASED ON SIDH

Burmester and Desmedt (Burmester and Desmedt, 1994; Burmester and Desmedt, 2005) and Steiner *et al.* (Steiner et al., 1996) proposed several types of GKE protocol based on the 2-party Diffie-Hellman (DH) protocol. These protocols offer however no security against quantum computer based attacks. The authors of (Furukawa et al., 2018) proposed an efficient post-quantum GKE protocol based on SIDH, by modifying the *Broadcast GKA Protocol* proposed

Algorithm 1: Isogeny-based Star GKT Protocol.

1: Each user U_i selects $m_{U_i}, n_{U_i} \in_R \mathbb{Z}/l_U^{e_U}\mathbb{Z}$ and computes a tuple $\{E_{U_i}, \phi_{U_i}(P_C), \phi_{U_i}(Q_C)\}$, then sends it to U_1.
2: The chair C selects $m_C, n_C \in_R \mathbb{Z}/l_C^{e_C}\mathbb{Z}$, $t \in_R T$ and a conference session key $K \in_R \{0,1\}^k$. Then C computes $E_C, \phi_C(P_U), \phi_C(Q_U)$, $K_i = H_t(j(E_{CU_i}))$, $c_i = K_i \oplus K$.
3: The chair C sends a tuple $\{E_C, \phi_C(P_U), \phi_C(Q_U), t, c_i\}$ to each user U_i.
4: Each user U_i computes $K_i = H_t(j(E_{CU_i}))$, decrypts $K = c_i \oplus K_i$.

Algorithm 2: Isogeny-based Tree GKT Protocol.

1: Each user U_a selects

$$m_{U_a}, n_{U_a} \in_R \begin{cases} \mathbb{Z}/l_0^{e_0}\mathbb{Z} & (\text{when } d = \lfloor \log a \rfloor \text{ is even}), \\ \mathbb{Z}/l_1^{e_1}\mathbb{Z} & (\text{when } d = \lfloor \log a \rfloor \text{ is odd}). \end{cases}$$

and computes a tuple

$$\begin{cases} \{E_{U_a}, \phi_{U_a}(P_1), \phi_{U_a}(Q_1)\} & (\text{when } d = \lfloor \log a \rfloor \text{ is even}), \\ \{E_{U_a}, \phi_{U_a}(P_0), \phi_{U_a}(Q_0)\} & (\text{when } d = \lfloor \log a \rfloor \text{ is odd}). \end{cases}$$

and then sends it to $U_{\lfloor a/2 \rfloor}, U_{2a}, U_{2a+1}$.
2: Each U_a computes $K_{a,\lfloor a/2 \rfloor} = H_{t_c}(j(E_{U_a U_{\lfloor a/2 \rfloor}}))$, $K_{a,2a} = H_{t_c}(j(E_{U_a U_{2a}}))$, and $K_{a,2a+1} = H_{t_c}(j(E_{U_a U_{2a+1}}))$, where $t_c \in T$ is a fixed constant. The root U_1 selects a group session key $K \in_R \{0,1\}^k$ and then sends $c_2 = K_2 \oplus K$, $c_3 = K_3 \oplus K$ to U_2, U_3, respectively.
3: **for** $d = 1$ to $\log n$ **do** if $\lfloor \log a \rfloor = d$ then
4: U_a decrypts c_a to get the conference key $K = c_a \oplus K_{a,\lfloor a/2 \rfloor}$.
5: U_a sends $c_{2a} = K \oplus K_{2a}$ to U_{2a} and sends $c_{2a+1} = K \oplus K_{2a+1}$ to U_{2a+1}.
6: **end for**

by Burmester and Desmedt. There are differences between DH and SIDH we have to consider in order to construct GKE based on SIDH:

- (Space of Ephemeral Secrets): In SIDH, Alice chooses two ephemeral secrets m_A, n_A from $\mathbb{Z}/l_A^{e_A}\mathbb{Z}$, and Bob chooses two phemeral secrets m_B, n_B from $\mathbb{Z}/l_B^{e_B}\mathbb{Z}$. On the other hand, in the DH protocol, Alice and Bob choose their secrets r_A and r_B from the common space $\mathbb{Z}_q$.
- (Space of the Shared Key): In SIDH, the shared secret is $j(E_{AB}) \in \mathbb{F}_{p^2}$. On the other hand, in the DH protocol, the shared secret is $g^{r_A r_B} \in \mathbb{Z}_p$, which similar to the ephemeral secrets, is also a large positive integer.

Dealing with the differences above with the proper iterative selection of the users' ephemeral secret space, we modify the all types of the GKE protocols proposed in (Burmester and Desmedt, 1994) and (Kim et al., 2004), to post-quantum protocols based on SIDH: we consider *Isogeny-based Star GKT Protocol*, *Isogeny-based Tree GKT Protocol*, *Isogeny-based Tree GKA Protocol*, *Isogeny-based Broadcast GKA Protocol*, and *Isogeny-based Cyclic GKA Protocol*.

3.1 Isogeny-based Star GKT Protocol

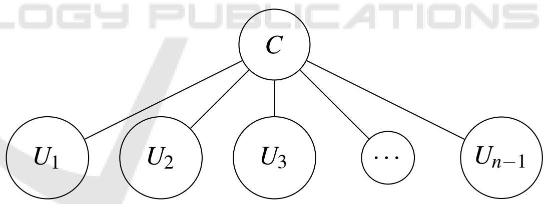

Figure 1: The system for isogeny-based Star GKT Protocol.

We describe the *Isogeny-based Star GKT Protocol*. This protocol is analogous to the Star based protocol proposed in (Burmester and Desmedt, 1994), based on the Diffie-Hellman key exchange. Algorithm 1 shows the algorithm of the protocol, and Figure 1 depicts the system. Notation used to denote the members and fixed public parameters is as follows:

- Members:
  - C (A chair)
  - U (Users): $\{U_1, \ldots, U_{n-1}\}$,
- Parameters:
  - $p = l_C^{e_C} l_U^{e_U} f \pm 1$, E_0, $\{P_C, Q_C\}$, $\{P_U, Q_U\}$.
  - Let $\mathcal{H} = \{H_t : t \in T\}$ be a hash function family indexed by a finite set T, where each H_t is a function from $\mathbb{F}_{p^2}$ to the key space $\{0,1\}^k$.

3.2 Isogeny-based Tree GKT Protocol

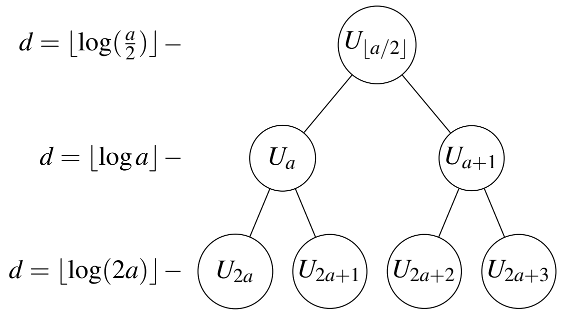

Figure 2: The system for the Isogeny-based Tree GKT Protocol.

We describe the *Isogeny-based Tree GKT Protocol*. This protocol is analogous to the tree based protocol proposed in (Burmester and Desmedt, 1994) based on the Diffie-Hellman key exchange. Algorithm 2 shows the algorithm of the protocol, and Figure 2 depicts the system. Notation used to denote the members and fixed public parameters is as follows:

- Users: $\{U_1, \ldots, U_n\}$. (U_1 is the root.)
- Parameters:
  - $p = l_0^{e_0} l_1^{e_1} f \pm 1$, E_0, $\{P_0, Q_0\}$, $\{P_1, Q_1\}$.
  - Let $\mathcal{H} = \{H_t : t \in T\}$ be a hash function family indexed by a finite set T, where each H_t is a function from $\mathbb{F}_{p^2}$ to the key space $\{0,1\}^k$.

3.3 Isogeny-based Tree GKA Protocol

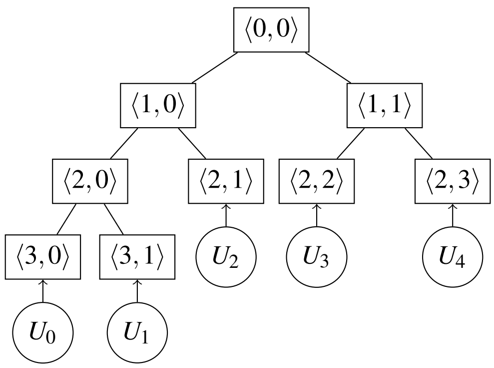

Figure 3: An example of the nodes tree for the isogeny-based tree GKA protocol ($n = 5$).

We describe an *Isogeny-based Tree GKA Protocol*. This protocol is analogous to the TGDH proposed in (Kim et al., 2004). Algorithm 3 shows the algorithm of the protocol, and Figure 3 depicts the system. The notation used for the members and fixed public parameters is as follows:

- Users: $\{U_0, \ldots, U_{n-1}\}$.
- Nodes: $\langle u, v \rangle$. ($2^{u_{\max}-1} \leq n \leq 2^{u_{\max}}$).
- Sponsor user(s): Every node has sponsor user(s), which we denote as $S_{\langle u,v \rangle}$.
- Parameters:
  - $p = l_0^{e_0} l_1^{e_1} f \pm 1$, E_0, $\{P_0, Q_0\}$, $\{P_1, Q_1\}$.
  - Let $\mathcal{H}^0 = \{H_t^0 : t \in T\}$ be a hash function family indexed by a finite set T, where each H_t^0 is a function from $\mathbb{F}_{p^2}$ to the (twin) space of ephemeral secrets $\mathbb{Z}/l_0^{e_0}\mathbb{Z} \times \mathbb{Z}/l_0^{e_0}\mathbb{Z}$.
  - Let $\mathcal{H}^1 = \{H_t^1 : t \in T\}$ be a hash function family indexed by a finite set T, where each H_t^1 is a function from $\mathbb{F}_{p^2}$ to the (twin) space of ephemeral secrets $\mathbb{Z}/l_1^{e_1}\mathbb{Z} \times \mathbb{Z}/l_1^{e_1}\mathbb{Z}$.
  - Let $\mathcal{H}' = \{H_t' : t \in T\}$ be a hash function family indexed by a finite set T, where each H_t' is a function from $\mathbb{F}_{p^2}$ to the key space $\{0,1\}^w$

3.4 Isogeny-based Broadcast GKA Protocol

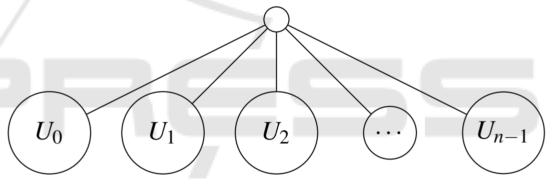

Figure 4: The system for Isogeny-based Broadcast GKA Protocol.

This protocol is proposed in (Furukawa et al., 2018), which is analogous to the broadcast protocol proposed in (Burmester and Desmedt, 1994) based on the Diffie-Hellman key exchange protocol. Algorithm 4 shows the algorithm of the protocol, and Figure 4 depicts the system. Notation used for members and the fixed public parameters is as follows:

- User: $U_0, \ldots, U_{n-1}$ (For simplicity, n is even)
- $p = l_0^{e_0} l_1^{e_1} f - 1$
- $\{P_0, Q_0\}$ is the basis of $E[l_0^{e_0}]$ and $\{P_1, Q_1\}$ is the basis of $E[l_1^{e_1}]$.
- define ι as follows;

$$\iota = \iota(i) := \begin{cases} 0 & \text{(when } i \text{ is even)}, \\ 1 & \text{(when } i \text{ is odd)}. \end{cases}$$

- index i is always calculated modulo n

Algorithm 3: Isogeny-based Tree GKA Protocol.

1: **for** $u = u_{\max}$ to 1 **do**

2: Denote $\iota = v \pmod 2$, and let $t_c \in T$ be a fixed constant. Each $S_{\langle u,v\rangle}$

$$\begin{cases} \text{selects } m_{\langle u,v\rangle}, n_{\langle u,v\rangle} \in_R \mathbb{Z}/l_\iota^{e_\iota}\mathbb{Z} & \text{(if } \langle u,v\rangle \text{ is a leaf),} \\ \text{computes } (m_{\langle u,v\rangle} || n_{\langle u,v\rangle}) = H^\iota_{t_c}(K_{\langle u,v\rangle}) & \text{(otherwise),} \end{cases}$$

and computes a tuple $\{E_{S_{\langle u,v\rangle}}, \phi_{S_{\langle u,v\rangle}}(P_\iota), \phi_{S_{\langle u,v\rangle}}(Q_\iota)\}$ and sends it to $S'_{\langle u,v\rangle}$, where $S'_{\langle u,v\rangle}$ is the sponsor of the node whose parent is common with $S_{\langle u,v\rangle}$.

3: Note that the parent node of $\langle u,v\rangle$ is $\langle u-1, \lfloor v/2\rfloor\rangle$. Each $S_{\langle u,v\rangle}$ and $S'_{\langle u,v\rangle}$ obtain $K_{\langle u-1,\lfloor v/2\rfloor\rangle} = j(E_{S_{\langle u,v\rangle}S'_{\langle u,v\rangle}})$. Then, let $S_{\langle u,v\rangle}$ and $S'_{\langle u,v\rangle}$ be $S_{\langle u-1,\lfloor v/2\rfloor\rangle}$.

4: **end for**

5: From the above steps, every user obtains $K_{\langle 0,0\rangle} = j(E_{S_{\langle 1,0\rangle}S_{\langle 1,1\rangle}})$, and then computes the conference key $K = H'_{t_c}(j(E_{S_{\langle 1,0\rangle}S_{\langle 1,1\rangle}}))$

Algorithm 4: Isogeny-based Broadcast GKA Protocol.

1: Every U_i randomly chooses $m_i, n_i \in (\mathbb{Z}/l_\iota^{e_\iota}\mathbb{Z})$. Then, U_i computes a tuple $\{E_{U_i}, \phi_{U_i}(P_{\iota+1}), \phi_{U_i}(Q_{\iota+1})\}$ and sends it to U_{i-1} and U_{i+1}.

2: Every U_i computes $j(E_{U_{i-1}U_i})$, and $j(E_{U_iU_{i+1}})$. Then, Every U_i broadcasts $X_i := j(E_{U_iU_{i+1}}) \cdot j(E_{U_{i-1}U_i})^{-1}$.

3: Every U_i calculates $K_i := j(E_{U_{i-1}U_i})^n \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdot \cdots \cdot X_{i-2}$. By simple arithmetic, for all i,

$$K = K_i = j(E_{U_1U_2}) \cdot j(E_{U_2U_3}) \cdot \cdots \cdot j(E_{U_nU_1}).$$

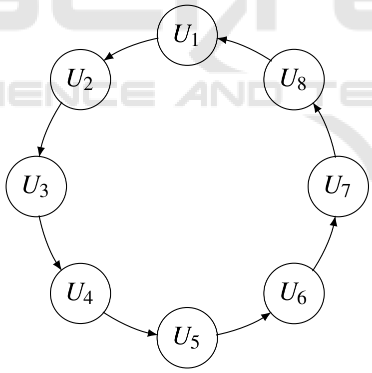

Figure 5: An example of the system for the Isogeny-based Cyclic GKA Protocol (when $n = 8$).

3.5 Isogeny-based Cyclic GKA Protocol

We describe an *Isogeny-based Cyclic GKA Protocol*. This protocol is analogous to the Cyclic protocol proposed in (Burmester and Desmedt, 1994) based on the Diffie-Hellman key exchange protocol. Algorithm 5 shows the algorithm of the protocol, and Figure 5 depicts the system. Notation used for members and fixed public parameters is the same as for the broadcast protocol.

4 SECURITY

In this section we give a brief sketch of the security reductions, relating the security of GKE protocols to the hardness of the appropriate underlying isogeny computation problem.

The security of GKT protocols, i.e. the isogeny-based star GKT protocol and the isogeny-based tree GKT protocol, follows straightforward from SIDH, of which security proof is given in (Jao and De Feo, 2011). Therefore, we focus on the security of the GKA protocols, i.e. the isogeny-based tree GKA protocol, the isogeny-based broadcast GKA protocol, and the isogeny-based cyclic GKA protocol.

Burmester and Desmedt gave a security proof of the broadcast protocol based on the Diffie-Hellman key exchange in (Burmester and Desmedt, 1994). Similarly, Furukawa *et al.* gave a security proof of the isogeny-based broadcast protocol based on SIDH in (Furukawa et al., 2018). The security of the isogeny-based tree GKA protocol, and isogeny-based cyclic GKA protocol described in this paper can be proven in a similar manner.

To start, assume the notation as follows:

- $\mathcal{E}_{SS,p}$: set of isomorphism classes of super singular EC defined on $\mathbb{F}_{p^2}$

Algorithm 5: Isogeny-based Cyclic GKA Protocol.

Every U_i randomly chooses $m_i, n_i \in (\mathbb{Z}/l_\iota^{e_\iota}\mathbb{Z})$. Then, U_i computes a tuple $\{E_{U_i}, \phi_{U_i}(P_{\iota+1}), \phi_{U_i}(Q_{\iota+1})\}$ and sends it to U_{i-1} and U_i.

Every U_i computes $j(E_{U_{i-1}U_i})$, and $j(E_{U_iU_{i+i}})$. Then, Every U_i computes $X_i := j(E_{U_iU_{i+1}}) \cdot j(E_{U_{i-1}U_i})^{-1}$. Let $b_0 = c_0 = 1$

for $i = 1$ to n **do** U_i computes (b_i, c_i), where $b_i = X_1 \cdot X_2 \cdots X_i$, $c_i = X_1^{i-1} \cdot X_2^{i-2} \cdots X_{i-1} = b_{i-1} \cdot c_{i-1}$, and sends them to U_{i+1}

end for

Let $d_0 = c_n = X_1^{n-1} \cdot X_2^{n-2} \cdots X_{n-1}$.

for $i = 1$ to n **do** U_i computes $d_i = b_n \cdot d_{i-1} \cdot X_i^{-n} = X_{i+1}^{n-1} \cdot X_{i+2}^{n-2} \cdots X_{i-1}$ and sends (b_n, d_i) to U_{i+1}.

end for

Every U_i calculates $K_i := j(E_{U_{i-1}U_i})^n \cdot d_i$ By simple arithmetic, for all i,

$$K = K_i = j(E_{U_1U_2}) \cdot j(E_{U_2U_3}) \cdot \cdots \cdot j(E_{U_nU_1}).$$

Table 1: Protocol comparison. Communication and computation are considered per user. I and H means the calculation cost of the isogeny map (ϕ_A or ϕ_B) and Hash function, respectively. M means the cost of multiplication of elements in $\mathbb{F}_{p^2}$.

Protocol	type	Communication (per user)	Rounds	Computation (per user)
Star GKT	Transfer	(chair): $2(n-1)$ (users): 2	2	(chair): $nI + H$ (users): $2I + H$
Tree GKT	Transfer	5	$1 + \lceil \log n \rceil$	$4I + 3H$
Tree GKA	Agreement	$1 + \lceil \log n \rceil$	$1 + \lceil \log n \rceil$	$\lceil \log n \rceil I + \lceil \log n \rceil H$
Broadcast GKA	Agreement	2	2	$3I + \frac{n(n+1)}{2}M$
Cyclic GKA	Agreement	6	$2n+1$	$3I + \frac{n(n+1)}{2}M$

- $\mathcal{J}_{SS,p} := \{ j \in \mathbb{F}_{p^2} \mid \exists E \in \mathcal{E}_{SS,p}, j = j(E) \}$
- $\mathcal{J}^n := \{ j_1 \cdots j_n \mid i \in [n], j_i \in \mathcal{J}_{SS,p} \} \subset \mathbb{F}_{p^2}$

The definition of hard problem and security are as follows:

Definition 1. *Super Singular Decisional Diffie-Hellman (SSDDH) problem is to distinguish the distribution of*

$$(E, E_A, \phi_A(P_B), \phi_A(Q_B), E_B, \phi_B(P_A), \phi_B(Q_A), j(E_{AB}))$$

and

$$(E, E_A, \phi_A(P_B), \phi_A(Q_B), E_B, \phi_B(P_A), \phi_B(Q_A), j).$$

Definition 2. *When any probabilistic polynomial algorithm $\mathcal{A}$ cannot distinguish K from a random element of $\mathcal{J}^n$, the isogeny-based broadcast protocol provides* **secrecy**.

Then, the statement of the theorem is as follows:

Theorem 1. *Under the assumption that SSDDH holds, the isogeny-based broadcast protocol provides* **secrecy**.

Proof. (sketch) Assume that the isogeny-based broadcast protocol does not provide secrecy: then there is a probabilistic polynomial algorithm $\mathcal{A}$ that can distinguish whether K is the shared key or a random element. One can then show that the SSDDH problem can be solved using $\mathcal{A}$; that is, we obtain a sample $(E, E_{A_1}, \{\phi_{A_1}(P_2), \phi_{A_1}(Q_2)\}, E_{A_n}, \{\phi_{A_n}(P_1), \phi_{A_n}(Q_1)\}, j)$ from the oracle of SSDDH, and distinguish whether $j = j(E_{A_1A_n})$ using the $\mathcal{A}$.

Indeed, from the sample above, we can calculate the sample for $\mathcal{A}$:

$$(E, \{E_{A_1}\}_{i=1}^n, \{\phi_{A_i}(P_{\iota-1}), \phi_{A_i}(Q_{\iota-1})\}_{i=1}^n, \{\phi_{A_i}(P_{\iota+1}), \phi_{A_i}(Q_{\iota+1})\}_{i=1}^n, \{X_i\}_{i=1}^n)$$

and calculate K. Then,

$$\begin{cases} \text{if } j = j(E_{A_1A_n}), & K \text{ is true shared key.} \\ \text{otherwise,} & K \text{ is random element.} \end{cases}$$

Thus, $\mathcal{A}$ can distinguish whether $j = j(E_{A_1A_n})$. □

5 COMPARISON

In this section, we compare theoretical and experimental costs of the five protocols described in Section 3.

Table 1 shows the theoretical costs of the GKE protocols. To compare the actual performance, we implemented the five protocols. We conducted

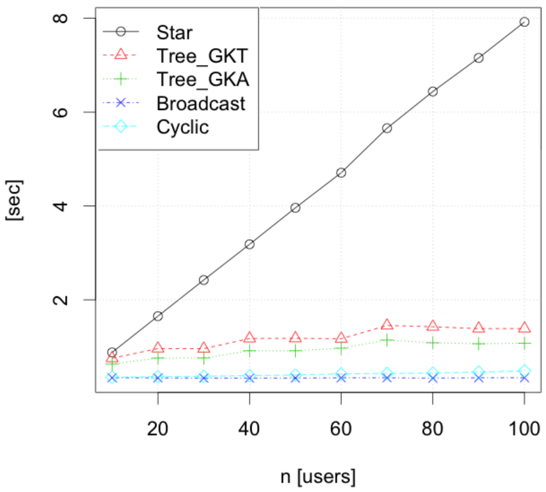

Figure 6: Experimental performance results (seconds).

experiments in one ordinary desktop PC (iMac 2017, 3.4 GHz Intel Core i5) and simulated the total computation time, from the time when the GKE runs to the time when all members obtain the group key. Note that the results do not include the time for communication. We show the experimental timing results in Figure 6.

6 CONCLUSION

We described five types of post-quantum GKE protocols based on SIDH. They were defined by modifying the classical GKE protocols based on Diffie-Hellman key exchange proposed by Burmester and Desmedt (Star, Broadcast and Cyclic GKE) and Kim *et al.* (Tree GKE). We theoretically analysed the computational costs, and also measured their experimental costs with a simple implementation. The results of our simulation indicate that all protocols, with exception of the isogeny-based star GKT protocol, are feasible in only 2 seconds for $n = 10, 20, \ldots, 100$ users. The experiments also confirms that the isogeny-based broadcast GKA protocol is the most efficient (it takes less than 0.5 seconds in our experiments).

REFERENCES

Barnes, R., Beurdouche, B., Millican, J., Omara, E., Cohn-Gordon, K., and Robert, R. (2020). The Messaging Layer Security (MLS) Protocol. Internet-Draft draft-ietf-mls-protocol-11, Internet Engineering Task Force. Work in Progress.

Bellare, M., Singh, A. C., Jaeger, J., Nyayapati, M., and Stepanovs, I. (2017). Ratcheted encryption and key exchange: The security of messaging. In Katz, J. and Shacham, H., editors, *CRYPTO 2017*, pages 619–650.

Burmester, M. and Desmedt, Y. (1994). A secure and efficient conference key distribution system. In *Workshop on the Theory and Application of Cryptographic Techniques*, pages 275–286. Springer.

Burmester, M. and Desmedt, Y. (2005). A secure and scalable group key exchange system. *Information Processing Letters*, 94(3):137–143.

Childs, A., Jao, D., and Soukharev, V. (2014). Constructing elliptic curve isogenies in quantum subexponential time. *Journal of Mathematical Cryptology*, 8:1–29.

Cohn-Gordon, K., Cremers, C., and Garratt, L. (2016). On post-compromise security. In *2016 IEEE 29th Computer Security Foundations Symposium (CSF)*, pages 164–178.

Cohn-Gordon, K., Cremers, C., Garratt, L., Millican, J., and Milner, K. (2018). On ends-to-ends encryption: Asynchronous group messaging with strong security guarantees. In *CCS '18*, pages 1802–1819. ACM.

Furukawa, S., Kunihiro, N., and Takashima, K. (2018). Multi-party key exchange protocols from supersingular isogenies. In *2018 International Symposium on Information Theory and Its Applications (ISITA)*, pages 208–212.

Herzberg, A. and Leibowitz, H. (2016). Can johnny finally encrypt?: Evaluating e2e-encryption in popular im applications. In *Proceedings of the 6th Workshop on Socio-Technical Aspects in Security and Trust*, STAST '16, pages 17–28. ACM.

Jao, D. and De Feo, L. (2011). Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Yang, B.-Y., editor, *Post-Quantum Cryptography*, pages 19–34.

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

Microsoft Community (2018). Skype insider preview, private conversations. https://answers.microsoft.com/en-us/skype/forum/skype_insiderms-skype_insnewsms.

National Institute of Standards and Technology (2020). Post-quantum cryptography. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography.

Open Whisper Systems (2018). Signal partners with microsoft to bring end-to-end encryption to skype. https://signal.org/blog/skype-partnership.

Perrin, T. and Marlinspike, M. (2016). The double ratchet algorithm. https://signal.org/docs/specifications/doubleratchet/doubleratchet.pdf.

Rösler, P., Mainka, C., and Schwenk, J. (2018). More is less: On the end-to-end security of group chats in Signal, WhatsApp, and Threema. In *2018 IEEE European Symposium on Security and Privacy (Euro S & P)*, pages 415–429.

Rostovtsev, A. and Stolbunov, A. (2006). Public-key cryptosystem based on isogenies. Cryptology ePrint Archive, Report 2006/145. https://eprint.iacr.org/2006/145.

Silverman, J. H. (1986). *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer.

Steiner, M., Tsudik, G., and Waidner, M. (1996). Diffie-hellman key distribution extended to group communication. CCS '96, pages 31–37. ACM.

Tanada, S., Suzuki, H., Naito, K., and Watanabe, A. (2016). Proposal for secure group communication using encryption technology. In *Mobile Computing and Ubiquitous Networking (ICMU), 2016 Ninth International Conference on*, pages 1–6. IEEE.