2008 Second IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2008)

Using non-commutative monoids to construct three-party key establishment

Song Han, Elizabeth Chang, Tharam S. Dillon

***Abstract*— Three-party key establishment protocol can help three participants to establish a shared secret key through interactions via public channels. In this paper, a novel three-party key agreement protocol is proposed. The protocol is based on non-commutative monoids in mathematics. It is a generic construction and one-time protocol per key establishment.**

***Index Terms*— Generic construction, key establishment, non-commutative monoid, one-time three-party key agreement, shared secret key**

I. INTRODUCTION

A Key establishment protocol is used to derive a shared secret by two or more parties as a function of information contributed by, or associated with, each of these, but no single party can predetermine the resulting value. Several key agreement protocols based on group theory have been proposed [1, 2, 3]. However, all these protocols are for two parties to establish a secret key. Those schemes cannot be transferred to three-party scenario in their present forms. On the other hand, for three-party key agreement, the probability of possible information leakage is larger than the one of two-party key agreement. This is because in the former case, the information amount transmitted between parties is much greater than that of the latter case. Therefore, it is interesting to propose a three-party key agreement protocol based on non-commutative monoids. In addition, the proposed key agreement should be immune from the existing attacks on algebraic method based cryptographic primitives.

In this paper, we will propose a generic three-party key agreement based on non-commutative monoids. The protocol is of one-time per key agreement. That means the secret keys as well as public keys of three participants are used only once for each key establishment. This can help to prevent from some existing attacks.

The organization of the rest of the paper is as follows: In the next section, some computational preliminaries are presented. Section III provides the proposed three-party key establishment scheme. Section IV analyses security discussion on the proposed scheme. The last section concludes this paper.

Song Han is with Curtin University of Technology. GPO Box U1987, Perth, WA 6845. Email: song.han@cbs.curtin.edu.au

Elizabeth Chang is with Curtin University of Technology. GPO Box U1987, Perth, WA 6845. Email: elizabeth.chang@cbs.curtin.edu.au

Tharam Dillon is with Curtin University of Technology. GPO Box U1987, Perth, WA 6845. Email: tharam.dillon@cbs.curtin.edu.au

II. PRELIMINARY

The mathematical definition for monoids will be reviewed in this section. In abstract algebra, a monoid is an algebraic structure with a single, associative binary operation and an identity element.

Definition A monoid is a set $\mathbf{M}$ with binary operation $* : \mathbf{M} \times \mathbf{M} \rightarrow \mathbf{M}$, obeying the following axioms:

- Associativity: for all $a, b, c \in \mathbf{M}$, $(a * b) * c = a * (b * c)$.
- Identity element: there exists an element $e \in \mathbf{M}$, such that for all $a \in \mathbf{M}$, $a * e = e * a = a$.
- Closure: for all $a, b \in \mathbf{M}$, $a * b$ is in $\mathbf{M}$. Alternatively, a monoid is a semigroup with an identity element.

A monoid satisfies all the axioms of a group with the exception of having inverses. A monoid with inverses is the same thing as a group.

Definition Submonoid: A submonoid of a monoid $\mathbf{M}$, is a subset $\mathbf{N}$ of $\mathbf{M}$ containing the unit element, and such that, if $x, y \in \mathbf{N}$, then $x * y \in \mathbf{N}$.

III. ONE-TIME THREE-PARTY KEY AGREEMENT BASED ON NON-COMMUTATIVE MONOIDS

Assume three participants Alice, Bob and Cindy will involve in the following protocol. Their unique

2008 Second IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2008)

means of communications is through public channels. Here *one-time* three-party key agreement indicates that the participants re-choose their secret keys for every time protocol run. This can help to prevent attacks from compromising possible long-term secret keys.

A. System setsup

Consider a 5-tuple: $(\mathbf{S}, \mathbf{T}, \alpha, f_1, f_2)$, where $\mathbf{S}$ and $\mathbf{T}$ are computable and non-commutative monoids. The three maps α, f_1 and f_2 are operations over $\mathbf{S}$ and $\mathbf{T}$ and defined as follows:

$$\alpha : \mathbf{S} \times \mathbf{S} \mapsto \mathbf{T}$$

$$f_1 : \mathbf{S} \times \mathbf{T} \mapsto \mathbf{T}$$

$$f_2 : \mathbf{S} \times \mathbf{T} \mapsto \mathbf{T}$$

They adhere to three axioms:

- Axiom 1: For all g, g_1, and $g_2 \in \mathbf{S}$, $\alpha(g, g_1 \cdot g_2) = \alpha(g, g_1) \cdot \alpha(g, g_2)$;
- Axiom 2: For any g, $h \in \mathbf{S}$, $f_1(g, \alpha(h, g)) = f_2(h, \alpha(g, h))$;
- Axiom 3: Given public elements g_1, g_2, ..., $g_n \in \mathbf{S}$, $h \in \mathbf{S}$ is a secret element, while

  $$\alpha(h, g_1), \alpha(h, g_2), ..., \alpha(h, g_n)$$

  are publicly known. Then, to determine h is not computable in polynomial time (i.e. it is infeasible in polynomial time).

Alice, Bob and Cindy will establish a shared secret key through running the following protocol. The n_1, n_2, and n_3 are three positive integers. We assume $\mathbf{S_A} \neq \mathbf{S_B} \neq \mathbf{S_C}$ for the following three monoids $\mathbf{S_A}$, $\mathbf{S_B}$, and $\mathbf{S_C}$.

1) Step 1.1: Alice is assigned a public monoid $\mathbf{S_A} \subsetneqq \mathbf{S}$. Suppose S_A is generated by the elements

   $$a_1, a_2, \cdots, a_{n_1}.$$

   That is, for any element $x \in \mathbf{S_A}$, x can be represented as $x = \prod_{i=1}^{n_1} a_i^{k(i)}$, where k_i $(1 \leq i \leq n_1)$ are non-negative integers .

2) Step 1.2: Bob is assigned a public monoid $\mathbf{S_B} \subsetneqq \mathbf{S}$. Suppose S_B is generated by the elements

   $$b_1, b_2, \cdots, b_{n_2}.$$

3) Step 1.3: Cindy is assigned a public monoid $\mathbf{S_C} \subsetneqq \mathbf{S}$. Suppose S_C is generated by the elements

   $$c_1, c_2, \cdots, c_{n_3}.$$

4) Step 1.4: Alice randomly chooses n_1 non-negative integers $e_1(1), e_1(2), \cdots, e_1(n_1)$ and computes

   $$a = \prod_{i=1}^{n_1} a_i^{e_1(i)}.$$

   Then $a \in \mathbf{S_A}$ (Alice keeps $e_1(i)(1 \leq i \leq n_1)$ privately). She then computes

   $$\alpha(a, b_1), \alpha(a, b_2), \cdots, \alpha(a, b_{n_2})$$

   and

   $$\alpha(a, c_1), \alpha(a, c_2), \cdots, \alpha(a, c_{n_3})$$

   Alice's secret key is a while her public key includes $\{\alpha(a, b_1), \alpha(a, b_2), \cdots, \alpha(a, b_{n_2})\}$ and $\{\alpha(a, c_1), \alpha(a, c_2), \cdots, \alpha(a, c_{n_3})\}$.

5) Step 1.5: Bob randomly chooses n_2 non-negative integers $e_2(1), e_2(2), \cdots, e_2(n_2)$ and computes

   $$b = \prod_{i=1}^{n_2} b_i^{e_2(i)}.$$

   Then $b \in \mathbf{S_B}$ (Bob keeps $e_2(i)(1 \leq i \leq n_2)$ privately). Bob then computes

   $$\alpha(b, c_1), \alpha(b, c_2), \cdots, \alpha(b, c_{n_3})$$

   and

   $$\alpha(b, a_1), \alpha(b, a_2), \cdots, \alpha(b, a_{n_1}).$$

   Bob's secret key is b while public key includes $\{\alpha(b, a_1), \alpha(b, a_2), \cdots, \alpha(b, a_{n_1})\}$ and $\{\alpha(b, c_1), \alpha(b, c_2), \cdots, \alpha(b, c_{n_3})\}$

6) Step 1.6: Cindy randomly chooses n_3 non-negative integers $e_3(1), e_3(2), \cdots, e_3(n_3)$ and computes

   $$c = \prod_{i=1}^{n_3} b_i^{e_3(i)}.$$

2008 Second IEEE International Conference on Digital Ecosystems and Technologies (IEEE DEST 2008)

Then $c \in \mathbf{S_C}$ (Cindy keeps $e_3(i)(1 \leq i \leq n_3)$ privately). She then computes

$$\alpha(c, a_1), \alpha(c, a_2), \cdots, \alpha(c, a_{n_1})$$

and

$$\alpha(c, b_1), \alpha(c, b_2), \cdots, \alpha(c, b_{n_2}).$$

Cindy's secret key is c while public key includes $\{\alpha(c, a_1), \alpha(c, a_2), \cdot \cdot \cdot, \alpha(c, a_{n_1})\}$ and $\{\alpha(c, b_1), \alpha(c, b_2), \cdots, \alpha(c, b_{n_2})\}$.

B. Shared key generation

Alice, Bob and Cindy share their public keys commonly. This can be achieved by publishing their public keys in a certified public key directory, e.g. a trusted public electronic board. They then follow the following steps to establish a shared secret key $F = f_1(a, \alpha(b, a)) \cdot f_2(c, \alpha(b, c)) \cdot f_2(a, \alpha(c, a)) \in \mathbf{T}$.

1) Step 2.1: With Alice and Bob's public keys, Cindy can use her secret key to compute

$$\alpha(b, c) = \prod_{i=1}^{n_3} \alpha(b, c_i)^{e_3(i)} \quad (1)$$

and

$$\alpha(a, c) = \prod_{i=1}^{n_3} \alpha(a, c_i)^{e_3(i)}. \quad (2)$$

Cindy then computes

$$\Omega_1 = f_2(c, \alpha(b, c))$$

and

$$\Omega_2 = f_1(c, \alpha(a, c)).$$

Finally, Cindy sends $\Omega_1 \cdot \Omega_2 \in \mathbf{T}$ to Alice and Bob.

2) Step 2.2: With Bob and Cindy's public keys, Alice can use her secret key to compute

$$\alpha(b, a) = \prod_{j=1}^{n_1} \alpha(b, a_j)^{e_1(j)} \quad (3)$$

and

$$\alpha(c, a) = \prod_{j=1}^{n_1} \alpha(c, a_j)^{e_1(j)}. \quad (4)$$

Alice then computes

$$\Omega_3 = f_1(a, \alpha(b, a))$$

and

$$\Omega_4 = f_2(a, \alpha(c, a)).$$

Finally, Alice sends $\Omega_3 \cdot \Omega_4 \in \mathbf{T}$ to Cindy and Bob.

3) Step 2.3: With Alice and Cindy's public keys, Bob can use his secret key to compute

$$\alpha(a, b) = \prod_{k=1}^{n_2} \alpha(a, b_k)^{e_2(k)} \quad (5)$$

and

$$\alpha(c, b) = \prod_{k=1}^{n_2} \alpha(c, b_k)^{e_2(k)}. \quad (6)$$

Bob then computes

$$\Omega_5 = f_2(b, \alpha(a, b))$$

and

$$\Omega_6 = f_1(b, \alpha(c, b)).$$

Finally, Bob sends $\Omega_5 \cdot \Omega_6 \in \mathbf{T}$ to Alice and Cindy.

The shared secret key of Alice, Bob and Cindy is $F = f_1(a, \alpha(b, a)) \cdot f_2(c, \alpha(b, c)) \cdot f_2(a, \alpha(c, a)) \in \mathbf{T}$.

IV. Security Analysis

We first explain why Alice, Bob and Cindy can have the shared secret key $F = f_1(a, \alpha(b, a)) \cdot f_2(c, \alpha(b, c)) \cdot f_2(a, \alpha(c, a)) \in \mathbf{T}$. Then we show an adversary cannot work out the shared secret key.

A. Correctness of the shared secret key

In fact,

- Alice has Ω_3 and $\Omega_1 \cdot \Omega_2$ and can compute $F_1 = \Omega_3 \cdot (\Omega_1 \cdot \Omega_2)$.
- Bob has Ω_5 and $\Omega_1 \cdot \Omega_2$ and can compute $F_2 = \Omega_5 \cdot (\Omega_1 \cdot \Omega_2)$.
- Cindy has Ω_2 and $\Omega_5 \cdot \Omega_6$ and can compute $F_3 = (\Omega_5 \cdot \Omega_6) \cdot \Omega_2$.

To explain Alice, Bob and Cindy share the key $F = f_1(a, \alpha(b, a)) \cdot f_2(c, \alpha(b, c)) \cdot f_2(a, \alpha(c, a))$, it is sufficient to show $F_1 = F_2 = F_3 = F$. In fact, by Axiom 2, we have

$$F_1 = \Omega_3 \cdot (\Omega_1 \cdot \Omega_2) \quad (7a)$$

$$= f_1(a, \alpha(b, a)) \cdot (f_2(c, \alpha(b, c)) \cdot f_1(c, \alpha(a, c))) \quad (7b)$$

$$= f_1(a, \alpha(b, a)) \cdot f_2(c, \alpha(b, c)) \cdot f_2(a, \alpha(c, a)) \quad (7c)$$

$$= F. \quad (7d)$$

$$F_2 = \Omega_5 \cdot (\Omega_1 \cdot \Omega_2) \tag{8a}$$
$$= f_2(b, \alpha(a,b)) \cdot (f_2(c, \alpha(b,c)) \cdot f_1(c, \alpha(a,c))) \tag{8b}$$
$$= f_1(a, \alpha(b,a)) \cdot f_2(c, \alpha(b,c)) \cdot f_f(a, \alpha(c,a)) \tag{8c}$$
$$= F. \tag{8d}$$

$$F_3 = (\Omega_5 \cdot \Omega_6) \cdot \Omega_2 \tag{9a}$$
$$= (f_2(b, \alpha(a,b)) \cdot f_1(b, \alpha(c,b))) \cdot f_1(c, \alpha(a,c)) \tag{9b}$$
$$= f_1(a, \alpha(b,a)) \cdot f_2(c, \alpha(b,c)) \cdot f_2(a, \alpha(c,a)) \tag{9c}$$
$$= F. \tag{9d}$$

B. Security of $\alpha(b,a)$ and $\alpha(c,a)$

Without Alice's private key, an adversary cannot compute $\alpha(b,a)$ and $\alpha(c,a)$ in polynomial time. This is because $\alpha(b,a) = \prod_{i=1}^{n_1} \alpha(b,a_i)^{e_1(i)}$ and $\alpha(c,a) = \prod_{i=1}^{n_1} \alpha(c,a_i)^{e_1(i)}$. Similarly, the security of $\alpha(a,b)$, $\alpha(c,b)$, $\alpha(a,c)$, and $\alpha(b,c)$ can be derived.

C. Security of $f_1(a, \alpha(b,a))$ and $f_2(a, \alpha(c,a))$

To identify the input a and $\alpha(b,a)$ to the function f_1 are both computably infeasible for an adversary in polynomial time. a is a secret key of Alice while the adversary cannot compute $\alpha(b,a)$ in polynomial time. Therefore, the adversary cannot work out $f_1(a, \alpha(b,a))$. Similarly, an adversary cannot work out $f_2(a, \alpha(c,a))$ and $f_2(c, \alpha(b,c))$ in polynomial time.

D. Security against existing attacks on algebraic based cryptographic primitives

Anshel et al. proposed a commutator key agreement protocol based on braid groups and their colored Burau representation [1]. Lee et al. proposed a summit set attack on Anshel et al.'s protocol [5]. In fact, the protocols in [1] which were broken by Lee et al. were only some instances of the key agreement based on braid groups. That attack could not be applied to the generic construction of Anshel et al.'s protocol [1]. Therefore, that attack could not be applied to our three-party key establishment either. This is because (1) our key agreement is a generic construction; (2) our key agreement is based on non-commutative monoids; (3) the key agreement is one-time per key establishment. In [4], Vasco et al. proposed two attacks on a public key cryptosystem based on free partially commutative monoids and groups. However, their attacks cannot be applied to our three-party key agreement protocol. This is because: On the one hand, their attacks are ciphertext only attacks and chosen ciphertext attacks while our protocol is key agreement. On the other hand, the monoids in our paper are assumed to be non-commutative.

Therefore, the adversary cannot compute the shared secret key

$$F = f_1(a, \alpha(b,a)) \cdot f_2(c, \alpha(b,c)) \cdot f_2(a, \alpha(c,a))$$

in polynomial time.

Remark: In this paper, we only consider the security of the protocol in polynomial time. This is reasonable because the secret keys of participants for one key agreement are used only once.

V. Conclusions and Future Work

In this paper we have proposed a three-party key agreement protocol. The protocol is novel because it is the first three-party key establishment based on non-commutative monoids. The purpose of the paper is to present a generic construction for designing three-party key agreement based on non-commutative monoids. Therefore, our next research is to give a concrete three-party key agreement protocol and show how exactly the parameters are to be chosen.

References

[1] I. Anshel, M. Anshel, B. Fisher, D. Goldfeld, New key agreement protocols in braid group cryptography, in CT-RSA 2001, Lecture Notes in Computer Science, vol. 2020, Springer, 2001.

[2] I. Anshel, M. Anshel, D. Goldfeld, An algegraic method for public key cryptography, Mathematical Research Letters 6 (1999) 287-291.

[3] I. Anshel, M. Anshel, D. Goldfeld, A method and apparatus for cryptographically secure algebraic key establishment protocols, International Patent Application Number: WO99/44324. US patent allowed. International Application Published Under the Patent Cooperation Treaty.

[4] M.I. Gonzalez Vasco, R. Steinwandt, Pitfalls in public key cryptosystems based on free partially commutative monoids and groups, Applied Mathematics Letters 19 (2006) 1037-1041.

[5] S.J. Lee, E. Lee, Potential weaknesses of the commutator key agreement protocol based on braid groups, in Eurocrypt2002, Lecture Notes in Computer Science, vol. 2332, Springer, 2002.