Computer Security Symposium 2021
26 - 29 October 2021

標準モデル安全な耐量子一方向匿名認証鍵交換

石橋　錬[1,a)]　米山　一樹[1]

概要：認証鍵交換 (AKE) は複数のパーティ間で共通のセッション鍵を共有するための暗号プロトコルである。通常の AKE ではセッション鍵の秘匿と相互認証が求められる。しかし，Tor や Riffle のような匿名ネットワークのように相互認証が望ましくない場合や，インターネットのようにユーザレベルでの証明書管理が不十分であるために相互認証の実現が困難な場合が存在する。Goldberg らは，クライアントだけがサーバを認証することでクライアントの匿名性を保証する一方向安全な AKE を定式化し，具体的な方式を提案した。しかし，既存の一方向安全な匿名 AKE 方式はランダムオラクルモデルで安全な方式しか知られていない。本稿では，ランダムオラクルモデルと標準モデルにおける一方向安全な匿名 AKE の一般構成をそれぞれ提案する。我々の一般構成によって，初めての標準モデルにおける同種写像問題に基づく一方向安全な耐量子匿名 AKE 方式を実現できる。

Post-Quantum One-Way Secure Anonymous Authenticated Key Exchange without Random Oracles

Ren Ishibashi[1,a)]　Kazuki Yoneyama[1]

Abstract: Authenticated Key Exchange (AKE) is a cryptographic protocol to share a common session key among multiple parties. Usually, PKI-based AKE schemes are designed to guarantee secrecy of the session key and mutual authentication. However, in practice, there are many cases where mutual authentication is undesirable, such as in anonymous networks like Tor and Riffle, or difficult to achieve due to the certificate management at the user level, such as the Internet. Goldberg et al. formulated the notion of one-way secure AKE, which guarantees the anonymity of the client by allowing only the client to authenticate the server, and proposed a concrete scheme. However, existing one-way secure anonymous AKE schemes are only known to be secure in the random oracle model. In this paper, we propose generic constructions of one-way secure anonymous AKE in the random oracle model and the standard model, respectively. Our constructions allow us to construct the first one-way secure post-quantum anonymous AKE scheme from isogenies in the standard model.

1. 背景

認証鍵交換 (AKE) はインターネットのような認証されていない通信路を用いて複数のパーティ間で共通のセッション鍵を共有するための暗号プロトコルである。通常の公開鍵基盤ベースの AKE では，各パーティは自分自身の static secret key (SSK) を保持し，SSK に対応する static public key (SPK) を発行する。SPK の正当性は認証局が発行する公開鍵証明書によって保証される。鍵交換のセッションでは，各パーティは ephemeral secret key (ESK) を生成し，ESK に対応する ephemeral public key (EPK) を相手に送信する。セッション鍵は，これらの鍵と鍵導出関数から導出される。通常の AKE はセッション鍵の秘匿と相互認証を目的としており，CK モデル [CK01] や eCK モデル [LLM07] などの安全性モデルにより証明可能安全性が定式化されている。

一方，Tor [DMS04] や Riffle [KLDF16] のような匿名ネットワークのように相互認証が望ましくない通信も普及しており，また，HTTPS のトランザクションでは，認証されていないクライアントが，認証されたサーバーと通信するのが一般的である。このように，クライアントが匿名であ

[1] 茨城大学
Ibaraki University
[a)] 21nm706r@vc.ibaraki.ac.jp

ることが望ましい場合や両側認証が必要ない場合が存在する。通常の AKE の安全性モデルでは，このような片側認証や匿名性を捉えることはできない。

片側認証を前提とし，匿名性を保証する暗号プロトコルとして一方向 AKE が知られている。一方向 AKE はクライアント・サーバの 2 者間通信を想定しており，サーバのみが SSK と SPK のペアを保持し，鍵交換セッションにおいてはクライアントとサーバの双方が ESK と EPK を生成することで，共通のセッション鍵を共有する。このように一方向 AKE は，クライアントに対する認証がない AKE である。Goldberg らは，匿名性を保証する一方向 AKE 安全性モデル [GSU12] を定式化し，具体的な方式を提案した。また，その他の一方向 AKE 安全性モデルとして，Dodis らのモデル [DF14] が提案されているが，彼らのモデルでは匿名性は保証していない。そのため，本研究では [GSU12] の安全性モデルに注目する。

現在，安全な一方向匿名 AKE 方式は，ランダムオラクルモデルにおいて安全な方式しか知られていない。また，量子計算機に対して安全な方式も提案されているが，部分的な状況だけしか耐量子安全性を保証できないという問題がある。本稿では，ランダムオラクルモデルと標準モデルにおける一方向匿名 AKE の一般構成を提案する。我々の一般構成は KEM に基づいており，適切な KEM 方式を用いることによって，初めての標準モデルにおいて安全な方式や，耐量子安全な方式を実現することができる。

2. 準備

2.1 一方向安全性モデル

本節では，Goldberg らによる安全性モデル [GSU12] を紹介する。彼らのモデルは片側認証におけるセッション鍵の秘匿性を捉えた一方向性 AKE 安全性と一方向性匿名性の定義からなる。

記法として $x \in_R X$ は要素 x を集合 X から一様ランダムに選択することを表す。

2.1.1 セッションと攻撃者の定義

[パーティ，鍵ペア]

パーティは確率的多項式時間チューリング機械としてモデル化される。各パーティは初期化メッセージを受信してアクティベートされ，プロトコルによって定義されたメッセージを返す。

各パーティが保持する鍵ペアを (x, X) という形式で表す。ただし，x は秘密値，X は公開値である。鍵ペアには，例えば公開鍵暗号における秘密鍵と公開鍵や，暗号化に用いる乱数と暗号文などが該当する。また，特定のセッションで使用する短期鍵ペアと全てのセッションで使用する長期鍵ペアの 2 種類があり，未使用の短期鍵ペアをオフラインで事前計算する場合もある。

各サーバは，SPK として公開値 X と ID ID_S を結合した証明書 $cert_X = (ID_S, X)$ を所有し，これをサーバ認証で使用する。あるパーティが公開値 X に対応する秘密値 x を所有している場合，そのパーティは公開値 X の owner であるという。

[プロトコル，セッション]

プロトコルの各実行はセッションと呼ばれ，各セッションにはパーティに割り当てられたセッション ID（sid）があり，各 sid はパーティ内で一意でなければならない。各セッションには，中間値を含むセッション状態が関連付けられており，パーティ U_P による sid のセッション状態を，$M^P_{state}[sid]$ によって示す。もし，セッション sid があるパーティ内で実行された場合，そのパーティを sid の owner と呼ぶ。また，あるセッションの owner がセッション鍵 sk を計算してセッションを終了した場合，そのセッションを completed session と呼ぶ。

[セッション実行]

パーティ U_P が owner であるセッション sid が complete すると，セッション内で使用された短期鍵ペア (x, X) は削除され，U_P はセッションの出力 $M^P_{out}[sid]$ として $\perp$ または $(sk, pid, \vec{v})$ を出力する。ただし，sk は鍵空間 $\mathcal{SK}$ に属するセッション鍵，pid は peer の ID または匿名シンボル $\circledast$，$\vec{v} = (\vec{v_0}, \vec{v_1}, ...)$ の各ベクトル $\vec{v_i}$ はセッションで使用された長期鍵と短期鍵の公開値のベクトルである。（例えば、$\vec{v_1}$ はパーティ U_1 が送信した公開値で構成されている値の集合である。）使用した公開値を出力の一部として含めることで，各セッションを一意に識別することができる。必要に応じて，$M^P_{out}[sid].sk$ という表記を使用して，セッション sid のセッション鍵を表す。その他の出力値についても同様に表す。

[攻撃者]

公開パラメータを $params$ とする。攻撃者 $\mathcal{A}$ は確率的多項式時間チューリング機械としてモデル化され，$params$ を入力とし，パーティ $P_1, ..., P_n$ にオラクルアクセスする。$\mathcal{A}$ はセッションのアクティブ化を含むユーザ間のすべての通信を制御する。$\mathcal{A}$ は以下の攻撃者用クエリを用いてパーティ P に特定の action を実行させる。

- $\mathsf{Send}(params, pid) \rightarrow (sid, msg)$：任意のパーティにセッションをアクティベートさせる。パーティは新しいセッションをアクティベートし，プロトコルに従ってメッセージを返す。入力値 $params$ はプロトコルによって定義され，次のようなものが含まれる。(1) 実行されるプロトコル，(2) パーティが自身の認証に使用する証明書，(3) セッション内の peer が使用する証明書。pid はセッションを確立する相手の ID であり，そのセッションが認証されていない匿名の peer とのセッションを想定している場合，pid は特別な記号「$\circledast$」となる。
- $\mathsf{Send}(sid, msg) \rightarrow msg'$：パーティは msg でセッショ

ン sid をアクティベートし，プロトコルに従ってメッセージ msg' を返す。

- RevealNext $\rightarrow X$：$\mathcal{A}$ はオフラインで事前計算された公開値を得る。パーティは新しい鍵ペア (x, X) を生成し，それを未使用として記録し，公開値 X を返す。
- Partner$(X) \rightarrow x$：$\mathcal{A}$ はプロトコルで使用された公開値 X に対応する秘密値 x を得る。もしパーティのメモリに鍵ペア (x, X) が記録されているなら，秘密値 x を返す。
- SessionKeyReveal$(sid) \rightarrow sk$：$\mathcal{A}$ は sid のセッション鍵を得る。sid のセッション鍵 $M_{out}^{P}[sid].sk$ を返す。

さらに，$\mathcal{A}$ は以下のクエリを用いて任意の公開鍵と証明書を生成することができる。

- EstablishCertificate(ID_i, X)：$\mathcal{A}$ は未使用の ID ID_i の公開値 X を含む証明書をすべてのパーティに登録する。$\mathcal{A}$ はその証明書の owner となる。あるパーティがこのクエリによって登録されたならば，そのユーザを dishonest と呼び，そうでなければ honest と呼ぶ。

曖昧性をなくすために，必要に応じてクエリ先を $\mathsf{Send}^{P_i}(sid, msg)$ のように上付き文字を用いて示す。

[**Pertnering**]

X がパーティ P_i への Send クエリの入力か RevealNext クエリの出力であり，かつ P_i に Partner クエリを発行していない場合を除き，攻撃者 $\mathcal{A}$ を値 X の partner と呼ぶ。あるパーティが $\mathcal{A}$ からのクエリ，または，セッションの実行によって鍵ペア (x, X) を生成した場合，そのパーティを X の partner と呼ぶ。また，もし異なる公開値 X と X' が同一の秘密値 x と対応する場合，$\mathcal{A}$ が X の partner であるならば、A はまた X' の partner であるとみなす。

[**正当性**]

以下の条件を満たしている場合，2 者間の鍵交換プロトコルは正当性を満たすという。

- 攻撃者 $\mathcal{A}$ は 2 者間で実行されるプロトコルのすべてのメッセージを改ざんせずに中継する。
- あるパーティが $pid \neq \circledast$ である Send クエリでアクティベートされた場合，そのパーティは pid の正しい証明書を所持する。

また，プロトコルの実行ごとに以下の条件が成り立つ必要がある。

- 両パーティは同じセッション鍵 sk と同じベクトル $\vec{v}$ を出力する。
- 各パーティの出力に含まれる値 pid が，そのパーティがアクティベートされる際に使用した Send クエリの pid と一致する。

2.1.2 一方向 AKE 安全性

セッションの freshness を次のように定義する。

[**freshness**]

以下の条件を満たす場合，パーティ P_i によるセッション sid は fresh であるという。

(1) $M_{out}^{P_i}$ の各ベクトル $\vec{v_j}$ に対して，$\vec{v_j}$ には $\mathcal{A}$ が partner でないような公開値 X が少なくとも 1 つ存在する。ただし，$j \geq 1$ である。

(2) $\mathcal{A}$ は任意のパーティ P_j に SessionKeyReveal(sid) クエリを発行していない。ただし P_j は，$M_{out}^{P_i}[sid].\vec{v} = M_{out}^{P_j}[sid].\vec{v}$ となるような $M_{out}^{P_i}[sid].pid$ の証明書の owner である。

一方向 AKE 安全性ゲームにおける攻撃者 $\mathcal{A}$ のゴールは，真のセッション鍵とランダムな鍵を識別することである。このゲームにおいて，$\mathcal{A}$ は特別なクエリとして以下のクエリを発行することができる。

- Test$(i, sid^*) \rightarrow \mathcal{SK}$：$sid^*$ は fresh なセッションでなければならない。$M_{out}^{P_i}[sid^*].sk = \perp$ または $M_{out}^{P_i}[sid^*].pid = \circledast$ なら abort する。もし $b = 0$ なら，$M_{out}^{P_i}[sid^*].sk$ を返す。そうでなければランダムな $\mathcal{SK}$ の要素を返す。このクエリは一度だけ発行できる。

一方向 AKE は片側認証であるため，テストセッション sid^* はサーバ認証を行うクライアント側のセッションのみを対象としている。攻撃者 $\mathcal{A}$ は sid^* のセッション鍵かランダムな鍵のいずれかをそれぞれ 1/2 の確率で得る。Test クエリ発行後も，$\mathcal{A}$ が受け取った鍵がランダムか否かを予想した結果 b' を出力するまでゲームは続行する。もし sid^* が最後まで fresh かつ，$\mathcal{A}$ の予想が正しい場合 (i.e., $b = b'$)，$\mathcal{A}$ はゲームに勝利したと定義する。

定義 2.1 (一方向 AKE 安全性) 一方向 AKE プロトコル Π に対する上記の攻撃ゲームにおける攻撃者 $\mathcal{A}$ の優位性を以下のように定義する。

$$Adv_{\Pi}^{1w-AKE}(\mathcal{A}) = \Pr[b = b'] - 1/2$$

セキュリティパラメータを κ とする。全ての確率的多項式時間攻撃者 $\mathcal{A}$ について，κ に対して Adv_{Π}^{1w-AKE} が無視できるならば，Π は一方向 AKE 安全であるという。

2.1.3 一方向匿名性

一方向匿名安全性ゲームにおける攻撃者 $\mathcal{A}$ のゴールは，2 つのパーティのうちどちらが鍵交換に参加しているかを識別することである。$\mathcal{A}$ が識別対象のパーティに直接クエリするのではなく，挑戦者 $\mathcal{C}$ がその通信を中継する。$\mathcal{A}$ は識別を試みる対象のパーティのインデックス i_0 と i_1 を $\mathcal{C}$ に与える。$\mathcal{C}$ はランダムに $i^* \in \{i_0, i_1\}$ を選択し，$\mathcal{A}$ と P_{i^*} の間のメッセージを中継する。$\mathcal{A}$ はこの i^* を予想する。

一方向匿名安全性ゲームにおいて，通常のクエリに加えて $\mathcal{A}$ は特別なクエリとして以下のクエリを $\mathcal{C}$ に発行することができる。始めの 2 つのクエリはテストセッションのアクティベートと通信のためのクエリである。

- Start$^{\mathcal{C}}(i_0, i_1, params, pid) \rightarrow msg'$：もし $i_0 = i_1$ なら abort する。そうでなければ，$i^* \in_R \{i_0, i_1\}$ をセットし，$(sid^*, msg') \leftarrow \mathsf{Send}^{P_{i^*}}(params, pid)$ をクエリして msg' を返す。このクエリは一度だけ発行できる。

- $\mathsf{Send}^{\mathcal{C}}(msg) \rightarrow msg' : msg' \leftarrow \mathsf{Send}^{P_{i^*}}(sid^*, msg)$ をクエリして msg' を返す。

$\mathcal{A}$ が $\mathcal{C}$ にクエリすることができるその他のクエリは，テストセッションに関する情報の漏洩を行うものである。

- $\mathsf{RevealNext}^{\mathcal{C}} \rightarrow X : \mathsf{RevealNext}^{P_{i^*}}$ をクエリして返す。ただし，生成された公開値がテストセッション以外のセッションで使用されてはいけない。また，$\mathcal{A}$ が $\mathsf{RevealNext}^{P_{i^*}}$ をクエリして生成された公開値がテストセッションで使用されてはいけない。
- $\mathsf{SessionKeyReveal}^{\mathcal{C}}() \rightarrow sk : \mathsf{SessionKeyReveal}^{P_{i^*}}(sid^*)$ をクエリして sk を返す。
- $\mathsf{Partner}^{\mathcal{C}} \rightarrow x : \mathsf{Partner}^{P_{i^*}}(X)$ をクエリして x を返す。ただし，X は $\mathsf{Send}^{\mathcal{C}}$ クエリで返された値である。

定義 2.2 (一方向匿名性) セキュリティパラメータを κ，$n \geq 1$ とする。もし κ における全ての確率的多項式時間攻撃者 $\mathcal{A}$ について，以下のゲームに勝つ優位性 $Adv_{\Pi}^{1w-anon}(\mathcal{A}) = \Pr[i^* = i'] - 1/2$ が κ に対して無視できるならば，プロトコル Π は一方向匿名性であるという。

- $\mathsf{Expt}_{\Pi,\kappa,n}^{1w-anon}(\mathcal{A})$:
  - $params$，パーティ $P_1, \ldots, P_n$ を初期化する。
  - $i' \leftarrow \mathcal{A}^{P_1,\ldots,P_n,\mathcal{C}}(params)$ を出力する。
  - $\mathcal{A}$ が $\mathsf{Start}^{\mathcal{C}}(i_0, i_1, params, pid)$ クエリを発行し，挑戦者 $\mathcal{C}$ が i^* を選択したと仮定する。もし $i^* = i'$ かつ $\mathcal{A}$ のクエリが以下の制約を満たすならば，$\mathcal{A}$ はゲームに勝利する。
    - P_{i_0} と P_{i_1} に $\mathsf{SessionKeyReveal}$ をクエリしない。
    - $\mathcal{C}$ によって返された任意の公開値 X に対して，$\mathsf{Partner}(X)$ をクエリしない。
    - P_{i_0} と P_{i_1} に $\mathsf{Send}(sid^*, \cdot)$ をクエリしない。
    - P_{i_0} と P_{i_1} は sid^* のプロトコル実行中，pid の正しい証明書を所持している。

定義 2.2 の制約は，P_{i_0} と P_{i_1} にテストセッションに関する情報をクエリして P^{i^*} を自明に知ることができないようにするためである。例えば，もし $i^* = i_0$ ならば，$\mathsf{SessionKeyReveal}^{P_{i_0}}(sid^*)$ クエリは真のセッション鍵を返し，$\mathsf{SessionKeyReveal}^{P_{i_1}}(sid^*)$ クエリは，P_{i_1} は sid^* に参加していないため，$\perp$ を返す。よって $\mathcal{A}$ は自明に $i^* = i_0$ と識別できてしまう。つまり，一方向匿名性における主な制限は，P_{i_0} と P_{i_1} へのクエリは挑戦者 $\mathcal{C}$ を仲介してクエリしなければならないこと，テストセッションで使用された公開値はその他のセッションで使用してはならないことである。

2.2 KEM

KEM の定義を紹介する。

定義 2.3 (KEM) KEM は次のように 3 つの確率的多項式時間アルゴリズム $(\mathsf{KeyGen}, \mathsf{EnCap}, \mathsf{DeCap})$ から構成される。

- $(ek, dk) \leftarrow \mathsf{KeyGen}(1^{\kappa}, r_g)$: 鍵生成アルゴリズムは 1^{κ} と $r_g \in \mathcal{RS}_{\mathcal{G}}$ を入力とし，公開鍵と秘密鍵の鍵ペア (ek, dk) を出力する。ただし，κ はセキュリティパラメータ，$\mathcal{RS}_{\mathcal{G}}$ は乱数空間である。
- $(K, C) \leftarrow \mathsf{EnCap}(ek, r_e)$: カプセル化アルゴリズムは公開鍵 ek と $r_e \in \mathcal{RS}_{\mathcal{E}}$ を入力とし，セッション鍵 $K \in \mathcal{KS}$ と暗号文 $C \in \mathcal{CS}$ を出力する。ただし，$\mathcal{RS}_{\mathcal{E}}$ は乱数空間，$\mathcal{KS}$ はセッション鍵空間，$\mathcal{CS}$ は暗号文空間である。
- $K \leftarrow \mathsf{DeCap}(dk, C)$: カプセル開放アルゴリズムは秘密鍵 dk と暗号文 $C \in \mathcal{CS}$ を入力とし，セッション鍵 $K \in \mathcal{KS}$ を出力する。

ここで，任意の $\kappa \in \mathbb{N}$，任意の公開鍵と秘密鍵 $(ek, dk) \leftarrow \mathsf{KeyGen}(1^{\kappa}, r_g)$，任意の鍵と暗号文 $(K, C) \leftarrow \mathsf{EnCap}(ek, r_e)$ に対して，$K \leftarrow \mathsf{DeCap}(dk, C)$ を満たすものとする。

KEM に対する安全性の定義を紹介する。

定義 2.4 (安全性) 任意の PPT 攻撃者 $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ に対して，$Adv^{ind-cca} = |\Pr[(ek, dk) \leftarrow \mathsf{KeyGen}(1^{\kappa}; r_g); state \leftarrow \mathcal{A}_1^{\mathcal{O}}(ek); b \leftarrow_R \{0,1\}; (K_0^*, C_0^*) \leftarrow \mathsf{EnCap}(ek; r_e); K_1^* \in_R \mathcal{KS}; b' \leftarrow \mathcal{A}_2^{\mathcal{O}}(ek, (K_b^*, C_0^*), state); b' = b] - 1/2|$ が κ に対して無視できるならば，KEM は IND-CCA 安全である。ただし，$\mathcal{O}$ は復号オラクルである。

KEM が δ-min-entropy KEM であるとは，任意の秘密鍵，$(K, C) \leftarrow \mathsf{EnCap}(ek, r_e)$ で定義される K の分布 $\mathcal{D}_{\mathcal{KS}}$，公開情報の分布 $\mathcal{D}_{pub}$，乱数 $r_e \in \mathcal{RS}_{\mathcal{E}}$ に対し，$H_{\infty}(\mathcal{D}_{\mathcal{KS}}|\mathcal{D}_{pub}) \geq \delta$ が成り立つときをいう。ここで，H_{∞} は min エントロピーである。

2.3 PKIC-KEM

暗号文が公開鍵と独立に生成可能な KEM (PKIC-KEM) [Yon12] の定義を紹介する。

定義 2.5 (PKIC-KEM) PKIC-KEM は次のように 4 つの確率的多項式時間アルゴリズム $(\mathsf{wKeyGen}, \mathsf{wEnCapC}, \mathsf{wEnCapK}, \mathsf{wDeCap})$ から構成される。

- $(ek, dk) \leftarrow \mathsf{wKeyGen}(1^{\kappa}, r_g)$: 鍵生成アルゴリズムは 1^{κ} と $r_g \in \mathcal{RS}_{\mathcal{G}}$ を入力とし，公開鍵と秘密鍵の鍵ペア (ek, dk) を出力する。ただし，κ はセキュリティパラメータ，$\mathcal{RS}_{\mathcal{G}}$ は鍵生成アルゴリズムの乱数空間である。
- $C \leftarrow \mathsf{wEnCapC}(r_e)$: 暗号文生成アルゴリズムは $r_e \in \mathcal{RS}_{\mathcal{E}}$ を入力とし，暗号文 $C \in \mathcal{CS}$ を出力する。ただし，$\mathcal{RS}_{\mathcal{E}}$ は暗号化アルゴリズムの乱数空間，$\mathcal{CS}$ は暗号文空間である。
- $K \leftarrow \mathsf{wEnCapK}(ek, C, r_e)$: カプセル化アルゴリズムは公開鍵 ek と暗号文 $C \in \mathcal{CS}$，$r_e \in \mathcal{RS}_{\mathcal{E}}$ を入力とし，セッション鍵 $K \in \mathcal{KS}$ を出力する。ただし，$\mathcal{KS}$ はセッション鍵空間である。
- $K \leftarrow \mathsf{DeCap}(dk, C)$: カプセル開放アルゴリズムは秘密鍵

dk と暗号文 $C \in \mathcal{CS}$ を入力とし，セッション鍵 $K \in \mathcal{KS}$ を出力する。

PKIC-KEM に対する安全性の定義を紹介する。

定義 2.6 (安全性) 任意の PPT 攻撃者 $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ に対して，$Adv^{ind-cpa} = |\Pr[(ek, sk) \leftarrow \mathsf{wKeyGen}(1^\kappa; r_g); state \leftarrow \mathcal{A}_1(ek); b \leftarrow_R \{0,1\}; C_0^* \leftarrow \mathsf{wEnCapC}(r_e); K_0^* \leftarrow \mathsf{wEnCapK}(C_0^*; r_e); K_1^* \in_R \mathcal{KS}; b' \leftarrow \mathcal{A}_2(ek, (K_b^*, C_0^*), state); b' = b] - 1/2|$ が κ に対して無視できるならば，PKIC-KEM は IND-CPA 安全である。

また，PKIC-KEM の δ-min-entropy も KEM と同様に定義することができる。

3. 既存方式の問題点

現在，一方向匿名 AKE の既存方式として，DH 問題に基づく方式 [GSU12],[BKM12]，DH 問題と格子問題に基づく（部分的に）耐量子安全な方式 [GK15] が知られている。これらの既存方式には問題点が 3 つある。

まず，上記の 3 つの方式は全てランダムオラクルモデルで証明されている。標準モデルにおいても安全な方式が望ましい。

次に，[BKM12] と [GK15] の方式は，クライアントの ESK の両方が同時に漏れないという弱い freshness の下でしか安全でない方式であり，基となる [GSU12] の安全性モデルの freshness の下では安全でない方式となっている。本来の安全性モデルにおいても安全な方式が望ましい。

最後に，[GSU12] の方式は明示的な認証として MAC を用いている。しかし，安全性モデルでは明示的な認証を要求しておらず，方式の証明にも MAC は寄与していない。効率のため，非明示的な認証が望ましい。

4. 一般構成

本稿ではランダムオラクルモデルと標準モデルにおいて，KEM を用いた一般構成を提案する。ランダムオラクルモデルにおける一般構成は，OW-CCA 安全な KEM と OW-CPA 安全な PKIC-KEM を構成要素とし，標準モデルにおける一般構成は，IND-CCA 安全な KEM と IND-CPA 安全な PKIC-KEM を構成要素とする。我々の方式は [GSU12] の安全性モデルの全漏洩パターンに対して安全であり，標準モデルにおける初めての一方向匿名 AKE 方式を実現する。標準モデルにおける提案方式の略図を図 2 に示す。

4.1 構成のアイディア

方式の構成にあたって，[FSXY15] の CK+安全な相互認証 AKE 方式を一方のパーティの長期鍵を省いた方式とした場合，一見すると一方向安全性を満たすように見える。そのような FSXY ベースの方式の略図を図 1 に示す。なお、$(\mathsf{KeyGen}, \mathsf{EnCap}, \mathsf{DeCap})$ は IND-CCA 安全な KEM、$(\mathsf{wKeyGen}, \mathsf{wEnCapC}, \mathsf{wEnCapK}, \mathsf{wDeCap})$ は IND-CPA 安全な PKIC-KEM である。CK+安全性モデルでは，テストセッションの短期鍵の漏洩を許しており，短期鍵の漏洩に対して安全性を保証するテクニックとして，TPRF トリック [FSXY15] や NAXOS トリック [LLM07] が知られている。しかし，一方向安全な方式は片側認証であり，匿名性を保証する必要があるクライアントは長期鍵ペアを持つことはできず，クライアント側ではそれらのトリックを利用することができない。さらに，Goldberg らの安全性モデルにおける Partner クエリは，公開値 X に対する秘密値 x を reveal するクエリである。例えば，図 1 のようにサーバ側で TPRF トリックを使用して C_T を作成したとしても，秘密値であるトリックの出力 r_{ST} が reveal されてしまう。よってサーバ側でも TPRF トリックを使用することはできない。よって，freshness の定義により (C_C, C_T) や (ek_T, ek_S) の秘密値の組み合わせで reveal されてしまうため，この方式は破られてしまう。このことから FSXY ベースの方式での構成は困難である。

また，AKE の一方向安全性モデルでは NextReveal クエリのために，各セッションで両パーティが使用する短期鍵はオフラインで事前に生成できなければならない。そのため，図 1 のような構成では，サーバはクライアントのメッセージを受信してから EPK を生成する必要があるため，このような 2-pass の通信を必要とする IND-CPA 安全な KEM では一方向 AKE を構成することはできない。

最後に，今回の方式はクライアントの匿名性を保証した方式であるため，各セッションはクライアントからアクティベートする必要がある。

これらの点を考慮して我々は，IND-CCA 安全な KEM と IND-CPA 安全な PKIC-KEM を用いて一方向 AKE を構成する。PKIC-KEM は公開鍵と独立して暗号文を生成することができるため，AKE の各セッションで使用する短期鍵をオフラインで生成することができる。また，安全性モデルの freshness の定義より，匿名性を保証するクライアント側で使用する ESK が 1 つであれば，クライアント側での漏洩を考えなくてよい。しかし，方式としてはクライアント側で 2 種類の KEM を使用するため，2 種類の乱数が必要となる。そこで我々は 1 つの ESK を疑似ランダム関数に通して 2 種類の乱数を生成し，これらを基にそれぞれの KEM の暗号文を生成する。[*1] そして，暗号文生成に使用した 2 種類の乱数を削除することで，NextReveal クエリの対象を最初に生成した 1 つの ESK とすることができる。よって本方式で使用する秘密値の個数は，クライアント側で 1 つ (ESK)，サーバ側で 2 つ (ESK,SSK) となる。

[*1] このテクニックの具体例として，最初に 1 つの乱数から 2 つの出力を得るための PRF F' と，それぞれの KEM の乱数空間を値域とする PRF F'_0 と F'_1 を用いて，$(r_0||r_1) \leftarrow \mathsf{F}(ID_S, r)$ を $(\mathsf{F}'_0(ID_S, \mathsf{F}'('0', r))||\mathsf{F}'_1(ID_S, \mathsf{F}'('1', r)))$ とする。このようにして 2 種類の乱数を 1 つの乱数から生成する。

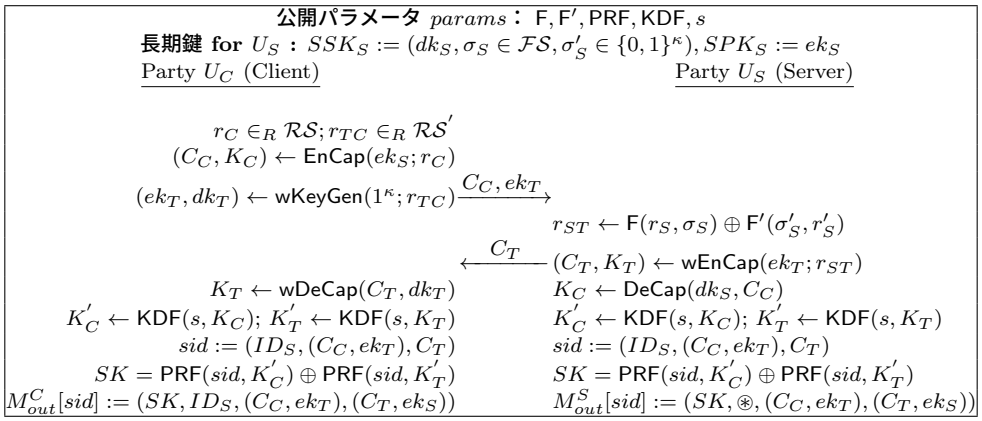

図 1 FSXY ベースの一方向 AKE 方式

提案方式の証明では，freshness の定義より，クライアント側の秘密鍵は reveal されないため，(1) サーバ側の SSK が reveal される場合と (2) サーバ側の ESK が reveal される場合を考慮する必要がある。(1) ではサーバ側の ESK は漏洩されないため，攻撃者は IND-CPA 安全な KEM の鍵である K_0 を計算することができない。同様に，(2) ではサーバ側の SSK は漏洩されないため，攻撃者は IND-CCA 安全な KEM の鍵である K_1 を計算することができない。したがって，提案方式は一方向安全性を満たす。また，クライアント側が各セッション内で使用する ESK は，クライアントの ID と独立した 1 つの乱数のみであるため，暗号文からクライアントに関する情報は一切得られない。よって，提案方式は一方向匿名性をもつ。

4.2 One-way AKE in Random Oracle Model

ランダムオラクルモデルによるプロトコルは OW-CCA 安全な KEM $(\mathsf{KeyGen}, \mathsf{EnCap}, \mathsf{DeCap})$ と OW-CPA 安全な PKIC-KEM $(\mathsf{wKeyGen}, \mathsf{wEnCapC}, \mathsf{wEnCapK}, \mathsf{wDeCap})$ から構成され，以下に示す通りである。

4.2.1 プロトコル

[Public Parameters]

κ をセキュリティパラメータとし，$\mathsf{H}_0 : \{0,1\}^* \to \mathcal{RS}_{CPA}$, $\mathsf{H}_1 : \{0,1\}^* \to \mathcal{RS}_{CCA}$，$\mathsf{H} : \{0,1\}^* \to \{0,1\}^\kappa$ をハッシュ関数とする。ただし，$\mathcal{RS}_{CPA}$ を OW-CPA 安全な KEM の乱数空間，$\mathcal{RS}_{CCA}$ を OW-CCA 安全な KEM の乱数空間とする。これらを公開パラメータの一部として与える。

[Secret and Public Keys]

パーティ U_S はランダムに $r \in_R \mathcal{RS}$ を選択して $(ek_S, dk_S) \leftarrow \mathsf{KeyGen}(1^\kappa; r)$ を実行し，U_S の証明書として $cert_{ek_S} = (ID_S, ek_S)$ をセットする。ただし，$\mathcal{RS}$ は KeyGen の乱数空間とする。パーティ U_S の長期鍵ペアは (ek_S, dk_S) である。

[Key Exchange]

長期鍵ペア (ek_S, dk_S) を持つパーティ U_S を server とし，パーティ U_C を client とする。U_C はクライアントとして初期化される際に U_S のサーバ証明書 $cert_{ek_S} = (ID_S, ek_S)$ を得る。

(1) U_C は $cert_{ek_S} = (ID_S, ek_S)$ を用いてサーバ検証を行う。U_C は未使用の鍵ペア (CT, r_C) を選択するか，ランダムに短期秘密鍵 $r_C \in_R \mathcal{RS}_\mathcal{E}$ を選択し，$r_0 \leftarrow \mathsf{H}_0(r_C)$，$r_1 \leftarrow \mathsf{H}_1(r_C)$ をセットする。また，$(C_1, K_1) \leftarrow \mathsf{EnCap}(ek_S; r_1)$，$C_0 \leftarrow \mathsf{wEnCapC}(r_0)$ を計算し，(r_0, r_1) を削除する。U_C は $CT = C_1 || C_0$ をセットし，U_S に (CT, ID_S) を送信する。

(2) U_S は (CT, ID_S) を受信すると，未使用の短期鍵ペア (ek_T, dk_T) を選択するか，ランダムに短期秘密鍵 $r_S \in_R \mathcal{RS}_\mathcal{G}$ を選択し，$(ek_T, dk_T) \leftarrow \mathsf{wKeyGen}(r_S)$ を実行して鍵ペアを生成して ek_T を U_C に送信する。U_S は $K_1 \leftarrow \mathsf{DeCap}(dk_S, C_1)$，$K_0 \leftarrow \mathsf{wDeCap}(dk_T, C_0)$ を計算し，$sid = (ID_S, CT, ek_T)$ をセットし，セッション鍵 $SK = \mathsf{H}(sid, K_0, K_1)$ を計算する。U_S は (r_S, dk_T) を削除し，$(SK, \circledast, (CT), (ek_T, ek_S))$ を出力する。

(3) U_C は ek_T を受信すると，$r_0 \leftarrow \mathsf{H}_0(r_C)$ をセットし，$K_0 \leftarrow \mathsf{wEnCapK}(ek_T, C_0; r_0)$ を計算する。U_C は $sid = (ID_S, CT, ek_T)$ をセットし，セッション鍵 $SK = \mathsf{H}(sid, K_0, K_1)$ を計算する。U_C は r_C, r_0 を削除し，$(SK, ID_S, (CT), (ek_T, ek_S))$ を出力する。

4.2.2 安全性

ランダムオラクルモデルによる提案方式の安全性を示す。証明の概要は 4.1 節を参照してほしい。

定理 4.1 $(\mathsf{KeyGen}, \mathsf{EnCap}, \mathsf{DeCap})$ が OW-CCA 安全な KEM，かつ $(\mathsf{wKeyGen}, \mathsf{wEnCap}, \mathsf{wDeCap})$ が OW-CPA 安全な PKIC-KEM，かつ $\mathsf{H}_0, \mathsf{H}_1, \mathsf{H}$ がランダムオラクルならば，構成した AKE 方式は一方向 AKE 安全なプロトコルである。

定理 4.2 ランダムオラクルモデルにおいて，構成した AKE 方式は一方向匿名性をもつ。

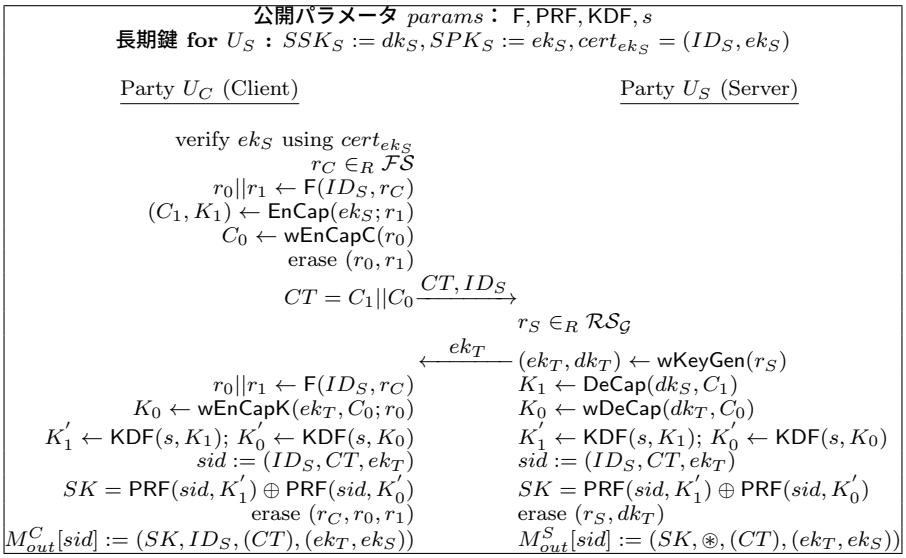

図 2 Generic construction in the standard model(GC-Std)

4.3 One-way AKE in Standard Model

標準モデルによるプロトコルは IND-CCA 安全な KEM $(\mathsf{KeyGen}, \mathsf{EnCap}, \mathsf{DeCap})$ と IND-CPA 安全な PKIC-KEM $(\mathsf{wKeyGen}, \mathsf{wEnCapC}, \mathsf{wEnCapK}, \mathsf{wDeCap})$ から構成され, 以下に示す通りである。

4.3.1 プロトコル

[Public Parameters]

κ をセキュリティパラメータとし，$\mathsf{F} : \{0,1\}^* \times \mathcal{FS} \to \mathcal{RS}_{CPA} \times \mathcal{RS}_{CCA}$, $\mathsf{PRF} : \{0,1\}^* \times \mathcal{FS} \to \{0,1\}^\kappa$ を疑似ランダム関数とする。また，$\mathsf{KDF} : Salt \times \mathcal{KS} \to \mathcal{FS}$ を鍵導出関数とし，$s \in_R Salt$ を選ぶ。ただし，$\mathcal{RS}_{CPA}$ を IND-CPA 安全な KEM の乱数空間，$\mathcal{RS}_{CCA}$ を IND-CCA 安全な KEM の乱数空間，$\mathcal{FS}$ を疑似ランダム関数の鍵空間 ($|\mathcal{FS}| = \kappa$)，$\mathcal{KS}$ を KEM のセッション鍵空間，$Salt$ を鍵導出関数のソルト空間とする。これらを公開パラメータの一部として与える。

[Secret and Public Keys]

パーティ U_S はランダムに $r \in_R \mathcal{RS}$ を選択して $(dk_S, ek_S) \leftarrow \mathsf{KeyGen}(1^\kappa; r)$ を実行し，U_S の証明書として $cert_{ek_S} = (ID_S, ek_S)$ をセットする。ただし，$\mathcal{RS}$ は KeyGen の乱数空間とする。パーティ U_S の長期鍵ペアは (dk_S, ek_S) である。

[Key Exchange]

長期鍵ペア (dk_S, ek_S) を持つパーティ U_S を server とし，パーティ U_C を client とする。U_C はクライアントとして初期化される際に U_S のサーバ証明書 $cert_{ek_S} = (ID_S, ek_S)$ を得る。

(1) U_C は $cert_{ek_S} = (ID_S, ek_S)$ を用いてサーバ検証を行う。U_C は未使用の短期鍵ペア (CT, r_C) を選択するか，ランダムに短期秘密鍵 $r_C \in_R \mathcal{FS}$ を選択し，$r_0||r_1 \leftarrow \mathsf{F}(ID_S, r_C)$ をセットする。また，$(C_1, K_1) \leftarrow \mathsf{EnCap}(ek_S; r_1)$，$C_0 \leftarrow \mathsf{wEnCapC}(r_0)$ を計算し，(r_0, r_1) を削除する。U_C は $CT = C_1||C_0$ をセットし，U_S に (CT, ID_S) を送信する。

(2) U_S は (CT, ID_S) を受信すると，未使用の短期鍵ペア (ek_T, dk_T) を選択するか，ランダムに短期秘密鍵 $r_S \in_R \mathcal{RS}_{\mathcal{G}}$ を選択し，$(ek_T, dk_T) \leftarrow \mathsf{wKeyGen}(r_S)$ を実行して鍵ペアを生成して ek_T を U_C に送信する。また，$K_1 \leftarrow \mathsf{DeCap}(dk_S, C_1)$，$K_0 \leftarrow \mathsf{wDeCap}(dk_T, C_0)$ を計算し，$K_1' \leftarrow \mathsf{KDF}(s, K_1)$，$K_0' \leftarrow \mathsf{KDF}(s, K_0)$ を計算する。U_S は $sid = (ID_S, CT, ek_T)$ をセットし，セッション鍵 $SK = \mathsf{PRF}(sid, K_1') \oplus \mathsf{PRF}(sid, K_0')$ を計算する。U_S は (r_S, dk_T) を削除し，$(SK, \circledast, (CT), (ek_T, ek_S))$ を出力する。

(3) U_C は ek_T を受信すると，$r_0||r_1 \leftarrow \mathsf{F}(ID_S, r_C)$ をセットし，$K_0 \leftarrow \mathsf{wEnCapK}(ek_T, C_0; r_0)$ を計算し，$K_1' \leftarrow \mathsf{KDF}(s, K_1)$，$K_0' \leftarrow \mathsf{KDF}(s, K_0)$ を計算する。U_C は $sid = (ID_S, CT, ek_T)$ をセットし，セッション鍵 $SK = \mathsf{PRF}(sid, K_1') \oplus \mathsf{PRF}(sid, K_0')$ を計算する。U_C は (r_C, r_0, r_1) を削除し，$(SK, ID_S, (CT), (ek_T, ek_S))$ を出力する。

4.3.2 安全性

標準モデルによる提案方式の安全性を示す。証明の概要は 4.1 節を参照してほしい。

定理 4.3 $(\mathsf{KeyGen}, \mathsf{EnCap}, \mathsf{DeCap})$ が IND-CCA 安全かつ δ-min-entropy KEM，$(\mathsf{wKeyGen}, \mathsf{wEnCap}, \mathsf{wDeCap})$ が IND-CPA 安全かつ δ-min-entropy PKIC-KEM，F, PRF が疑似ランダム関数，KDF が鍵導出関数ならば，構成した AKE 方式は一方向 AKE 安全なプロトコルである。

定理 4.4 標準モデルにおいて，構成した AKE 方式は一方向匿名性をもつ。

function wKeyGen	function wEnCapC	function wEnCapK	function wDeCap
Input: $1^\kappa, r_g \in_R \mathcal{RS}_G$	**Input:** $r \in_R \mathcal{K}_2$	**Input:** pk_3, sk_2	**Input:** sk_3, C_0
$sk_3 \in_R \mathcal{K}_3$	$sk_2 \leftarrow r$	$j \leftarrow \mathsf{isoex}_2(pk_3, sk_2)$	$j \leftarrow \mathsf{isoex}_3(C_0, sk_3)$
$pk_3 \leftarrow \mathsf{isogen}_3(sk_3)$	$C_0 \leftarrow \mathsf{isogen}_2(sk_2)$	$K \leftarrow j$	$K \leftarrow j$
return: (pk_3, sk_3)	**return:** C_0	**return:** K	**return:** K

図 3 SIKE-PKE [Jao17] に基づく PKIC-KEM 方式

5. DH 問題に基づく実装方式

5.1 ランダムオラクルモデル

一般構成 GC-ROM を IND-CCA 安全な KEM である PSEC-KEM [Sho01] と，OW-CPA 安全であり PKIC-KEM である ElGamal KEM を用いて実装することで CDH 仮定に基づく方式が得られる。

5.2 標準モデル

一般構成 GC-Std を IND-CCA 安全な KEM である CS-KEM [CS98] と，IND-CPA 安全であり PKIC-KEM である ElGamal KEM を用いて実装することで DDH 仮定に基づく方式が得られる。この方式は標準モデルにおける初めての一方向匿名 AKE 方式である。

6. 同種写像問題に基づく実装方式

6.1 ランダムオラクルモデル

6.1.1 SIDH 方式

一般構成 GC-ROM を IND-CCA 安全な SIKE-KEM [Jao17] と，IND-CPA 安全な SIKE-PKE [Jao17] を用いて実装することで，SI-CDH 仮定に基づく方式が得られる。PKE を IND-CPA 安全な KEM 方式として使用するために，暗号化アルゴリズムの暗号文 c_1 の生成と復号化アルゴリズムの m の復号を無くし，$j \leftarrow \mathsf{isoex}_2(pk_3, sk_2)$ の j を KEM のセッション鍵とする。また，この KEM 方式を図 3 のように構成することで PKIC-KEM として使用することができる。

6.1.2 CSIDH 方式

一般構成 GC-ROM を IND-CCA 安全な CSIDH-PSEC-KEM [Yon21] と，IND-CPA 安全であり PKIC-KEM である CSIDH-KEM [CLM18] を用いて実装することで CSI-DDH 仮定に基づく方式が得られる。なお，CSIDH-KEM は図 3 と同様な方法で PKIC-KEM として利用できる。

6.2 標準モデル

6.2.1 CSIDH 方式

一般構成 GC-Std を IND-CCA 安全な KEM [AFMP20] と，IND-CPA 安全であり PKIC-KEM である CSIDH-KEM [CLM18] を用いて実装することで CSI-DDH 仮定と弱疑似ランダム効果的群作用 [AFMP20] に基づく方式が得られる。この方式は標準モデルにおける初めての耐量子一方向匿名 AKE 方式である。

参考文献

[GSU12] I. Goldberg, D. Stebila, B. Ustaoglu,: Anonymity and one-way authentication in key exchange protocols, *Designs, Codes and Cryptography*, pp. 1–25, 2012.

[Sho01] V. Shoup,: A Proposal for an ISO Standard for Public Key Encryption, *Cryptology ePrint Archive*, Report 2001/112, 2001.

[CS98] R. Cramer, V. Shoup,: Design and Analysis of Practical Public-key Encryption Schemes Secure against Adaptive Chosen Ciphertext Attack, *LNCS Springer Verlag*, vol. 1462, pp. 13–25, 1998.

[Jao17] D. Jao et al.,: Supersingular isogeny key encapsulation, Round 1 submission, *NIST Post-Quantum Cryptography Standardization*, 2017.

[Yon21] K. Yoneyama,: Post-Quantum Variants of ISO/IEC Standards: Compact Chosen Ciphertext Secure Key Encapsulation Mechanism from Isogenies, *IEICE Transactions on Fundamentals of Electronics*, Communications and Computer Sciences Vol.E104-A No.1, pp.69-78, 2021.

[Yon12] K. Yoneyama,: One-Round Authenticated Key Exchange with Strong Forward Secrecy in the Standard Model against Constrained Adversary, *IWSEC 2012*, pp. 69-86, 2012.

[FSXY15] A. Fujioka, K. Suzuki, K. Xagawa, K. Yoneyama,: Strongly secure authenticated key exchange from factoring, codes, and lattices, *Des. Codes Cryptogr*, pp. 469-504, 2015.

[CLM18] W. Castryck, T. Lange, C. Martindale et al.,: CSIDH: An efficient post-quantum commutative group action, *ASIACRYPT 2018*, LNCS Vol. 11274, pp. 395–427. Springer, 2018.

[AFMP20] N. Alamati, L. D. Feo, H. Montgomery, and S. Patranabis,: Cryptographic group actions and applications, *ASIACRYPT 2020*, 2020.

[BKM12] M. Backes, A. Kate, E. Mohammadi,: Ace: An Efficient Key-Exchange Protocol for Onion Routing, *11th ACM WPES*, pp. 55–64, 2012.

[GK15] S. Ghosh, A. Kate,: Post-quantum forward-secure onion routing (future anonymity in today' s budget), *Cryptology ePrint Archive*, Report 2015/008, 2015.

[CK01] R. Canetti, H. Krawczyk,: Analysis of key-exchange protocols and their use for building secure channels, *Eurocrypt 2001*, LNCS Vol. 2045, pp. 453-474. Springer, 2001.

[LLM07] B. LaMacchia, K. Lauter, A. Mityagin,: Stronger security of authenticated key exchange, *ProvSec 2007*, LNCS vol. 4784, pp. 1-16. Springer, 2007.

[DMS04] R. Dingledine, N Mathewson, P. Syverson,: Tor: The second-generation onion router, *13th USENIX Security Symposium*, 2004.

[KLDF16] A. Kwon, D. Lazar, S. Devadas, B. Ford,: Riffle: An efficient communication system with strong anonymity, *16th PETS*, 2016

[DF14] Y. Dodis, D. Fiore,: Interactive encryption and message authentication, *SCN 2014*, 2014.