Vol.2018-DPS-174 No.7
Vol.2018-CSEC-80 No.7
2018/3/5

HEMSにおける機器認証プロトコルの安全性証明（その２）

四方 順司[1]　打越 忠宏[2]　海老名 将宏[2]　佐藤 慎悟[2]　増田 洋一[3]　海上 勇二[3]　高添 智樹[3]

概要：近年, 家電機器や住宅設備機器において, 住宅のエネルギー管理システム HEMS (Home Energy Management System) が普及し, HEMS を構成する機器のネットワーク接続が広まっている. そのため, HEMS における機器認証プロトコルの安全性を解析・評価することは重要である. 著者らは, SCIS2018 において, エコーネットコンソーシアムにおいて検討されている機器認証プロトコルの主要部が, 外部攻撃者に対して安全であることを示した. 本稿では, さらに, 機器認証プロトコルの主要部以外のグループ鍵管理や鍵更新に関しても安全性評価を行い, また, 他グループの通信内容やグループ鍵を取得できるような内部攻撃者に対する安全性に関しても評価を行う.

キーワード：HEMS, IoT セキュリティ，機器認証プロトコル, 安全性証明

Security Proof of a Device Authentication Protocol for HEMS (Part II)

Junji Shikata[1]　Tadahiro Uchikoshi[2]　Masahiro Ebina[2]　Shingo Sato[2]　Yoichi Masuda[3]
Yuji Unagami[3]　Tomoki Takazoe[3]

1. はじめに

近年, 家電機器や住宅設備機器において, 住宅のエネルギー管理システム HEMS (Home Energy Management System) が普及し, HEMS を構成する機器のネットワーク接続が広まっている。日本国内においては, エコーネットコンソーシアムにて規格策定された ECHONET Lite が，2011 年度にスマートハウス標準化検討会（経済産業省）にて HEMS における公知な標準インターフェースとして推奨されたことをきっかけに，HEMS のアプリケーションプロトコルとして普及しつつある [10]．そのため, HEMS における機器認証プロトコルの安全性を事前に解析・評価しておくことは重要である．最近, PANA/EAP-TLS と IEEE 802.21 のグループ鍵管理を応用した HEMS 通信の機器認証プロトコルとグループ鍵管理およびマルチキャスト通信保護の手法が増田・小椋 [9] によって提案され, 現在, エコーネットコンソーシアムにおいて検討されている. また, Hanatani ら [5] は, IEEE 802.21 におけるグループ鍵管理およびマルチキャスト通信の安全性を示している. 一方, 著者らは, SCIS2018 において, エコーネットコンソーシアムにおいて検討されている機器認証プロトコルの主要部が, 外部攻撃者に対して安全であることを示した [8]. 本稿では, さらに, 機器認証プロトコルの主要部以外のグループ鍵管理や鍵更新に関しても安全性評価を行い, また, 他グループの通信内容やグループ鍵を取得できるような内部攻撃者に対する安全性に関しても評価を行う.

2. 準備

本節では，本稿で利用する暗号基礎技術について簡単に記述する．また，本稿では ϵ を negligible function とする．negligible であるとは，$^{\forall}c > 0, ^{\exists}N \in \mathbb{N}, ^{\forall}n \geq N$ に対して、$\epsilon = \epsilon(n) < \frac{1}{n^c}$ となる場合である．自然数 N に対して $[N] := \{1, 2, \ldots, N\}$ とする.

[1] 横浜国立大学大学院環境情報研究院, 横浜国立大学先端科学高等研究院. Graduate School of Environment and Information Sciences, Yokohama National University. Institute of Advanced Sciences, Yokohama National University.
[2] 横浜国立大学大学院環境情報学府. Graduate School of Environment and Information Sciences, Yokohama National University.
[3] パナソニック株式会社. Panasonic Corporation.

2.1 共通鍵暗号 (SKE: Symmetric Key Encryption)

共通鍵暗号 (SKE) は，以下の 2 つの多項式時間アルゴリズムから構成される．ここで，$\mathcal{K}_{ske}$ を SKE の鍵空間，$\mathcal{M}_{ske}$ を SKE のメッセージ空間とする．

- $e_M \leftarrow \mathrm{SKE.Enc}(K_{ske}, M)$: 共通鍵 $K_{ske} \in \mathcal{K}_{ske}$ とメッセージ $M \in \mathcal{M}_{ske}$ を入力とし，M の暗号文 e_M を出力する．
- $M \leftarrow \mathrm{SKE.Dec}(K_{ske}, e_M)$: K_{ske} と e_M を入力とし，e_M の復号結果を出力する．

ここで, 全ての $K_{ske} \in \mathcal{K}_{ske}$ と $M \in \mathcal{M}_{ske}$ に対して, $e_M \leftarrow \mathrm{SKE.Enc}(K_{ske}, M)$ ならば, 必ず $\mathrm{SKE.Dec}(K_{ske}, e_M) = M$ が成立する．

定義 1. 共通鍵暗号における *indistinguishability against chosen-plaintext attack(IND-CPA)* は，以下のように定義される．A は，*SKE=*{*SKE.Enc,SKE.Dec*} における多項式時間攻撃者とする．ここで，以下のゲームを考える．

Step 1: $b \leftarrow_R \{0,1\}, K_{ske} \leftarrow_R \mathcal{K}_{ske}$.

Step 2: $b' \leftarrow A^{O}(1^{\lambda})$.

ここで，O は共通鍵暗号の暗号化オラクルであり，メッセージ対 (M_0, M_1) を受け取り, M_b を暗号化して返すオラクルである．このとき，上記のゲームにおける A のアドバンテージは $\mathrm{Adv}^{ind-cpa}_{SKE,A} := \left|\Pr[b' = b] - \frac{1}{2}\right|$ という式で定義される．共通鍵暗号が *IND-CPA* 安全性を満たすとは，任意の多項式時間攻撃者 A に対して $\mathrm{Adv}^{ind-cpa}_{SKE,A} \leq \epsilon$ が成立する時である．

2.2 認証付き暗号 (AE: Authenticated Encryption)

認証付き暗号 (AE) は，以下の 2 つの多項式時間アルゴリズムから構成される．ここで，$\mathcal{K}_{ae}$ を AE の鍵空間，$\mathcal{M}_{ae}$ を AE のメッセージ空間とする．

- $C_M \leftarrow \mathrm{AE.Enc}(K_{ae}, M)$: 共通鍵 $K_{ae} \in \mathcal{K}_{ae}$ とメッセージ $M \in \mathcal{M}_{ae}$ を入力とし，M の暗号文 C_M を出力する．
- M or $\perp \leftarrow \mathrm{AE.Dec}(K_{ae}, C_M)$: K_{ae} と C_M を入力とし，C_M の復号結果を出力する．

ここで，全ての $K_{ae} \in \mathcal{K}_{ae}$,$M \in \mathcal{M}_{ae}$ に対して，$C_M \leftarrow \mathrm{AE.Enc}(K_{ae}, M)$ ならば，必ず $\mathrm{AE.Dec}(K_{ae}, C_M) = M$ が成立する．

定義 2. 認証付き暗号における *indistinguishability against chosen-ciphertext attack (IND-CCA)* は，以下のように定義される．A は，認証付き暗号
AE={*AE.Enc,AE.Dec*} に対する多項式時間攻撃者とする．

Step 1: $K_{ae} \leftarrow_R \mathcal{K}_{ae}$, $b \leftarrow_R \{0,1\}$, $S \leftarrow \emptyset$.

Step 2: $b' \leftarrow A^{O_1, O_2}(1^{\lambda})$.

ここで，O_1 は二つのメッセージ対 (M_0, M_1) を入力すると M_b を暗号化して返すオラクルであり，O_2 は暗号文 C_M を受け取るとその復号結果を返すオラクルである．それぞれ以下のように定義する．

O_1*:* $C_M \leftarrow AE.Enc(K_{ae}, M_b)$, $S \leftarrow S \cup \{C_M\}$. *Return* C_M.

O_2*: If* $C_M \in S$, *return* $\perp$. *Else* $M \leftarrow AE.Dec(K_{ae}, C_M)$, *return* M.

この時，上記のゲームにおける A のアドバンテージは $\mathrm{Adv}^{ind-cca}_{AE,A} := \left|\Pr[b' = b] - \frac{1}{2}\right|$ という式で定義される．認証付き暗号が *IND-CCA* 安全性を満たすとは，任意の多項式攻撃者 A に対して，$\mathrm{Adv}^{ind-cca}_{KS,A} \leq \epsilon$ となる時である．

2.3 セッション鍵共有プロトコル (SKS: Session Key Share Protocol)

いま，パーティの集合 $\mathcal{P} := \{P_1, ..., P_n\}$ を考える．任意のパーティ P_i は，セッション鍵共有プロトコル (SKS) の入力として (P_i, P_j, s) を入力する．ここで，P_j は他のパーティ，s はセッション識別子とする．SKS の入力に (P_i, P_j, s), (P_j, P_i, s') が入力され $s = s'$ である場合に，P_i と P_j はマッチングされ、パートナーと呼ばれる．マッチング後，パートナー間でメッセージのやり取りをし，最終的に SKS はローカル出力として，セッションのパートナー識別子，セッション識別子，セッション鍵を出力する．パーティ P_i において (P_i, P_j, s) がマッチングされている場合に，共有されるセッション鍵を $K^s_{P_i}$ とする．このとき，以下の攻撃モデルを考える．

定義 3. *(UM: Unauthenticated-Links Adversarial Model)[2] SKS* に対する多項式時間攻撃者を A とする．A はパーティ間の通信を盗聴，改ざんすることができるとする．このとき，攻撃者が発行できるクエリを以下のように定義する．

1. *Initialize*(P_i)*:* パーティ P_i を初期化し，初期化したセッション識別子 s を返す．
2. *Invoke*(P_i, P_j, s)*:* 初期化されたセッション s において，P_i と P_j でプロトコルを実行する．
3. *RevealState*(P_i, s)*:* A はあるパーティ P_i とセッション識別子 s を入力することで，まだ終了していないセッションのセッション状態を知ることができる．
4. *RevealKey*(P_i, s)*:* A は，パーティ P_i とセッション識別子 s を入力することで，指定されたセッションで生成されたセッション鍵を受け取る．
5. *Corrupt*(P_i)*:* A は，任意のパーティ P_i をコラプトすることができる．コラプトした場合，A は P_i の状態 *(*秘密情報を含む*)* を知ることができる．
6. セッション失効クエリ*:* A は，任意のパーティ P_i 内のすべての終了セッション (P_i, P_j, s) を失効させることができる．失効されたセッションは，セッション鍵 K が消去され失効ラベルを持つ．また，失効ラベルを持つセッションに対して，A は *RevealKey* クエリを発行することができず，*Corrupt* クエリを発行してもセッション鍵は入手できない．

また，パーティ P_i 内のセッション (P_i, P_j, s) に対して *3,4,5* いずれかのクエリが発行された場合に，(P_i, P_j, s) は *locally exposed* であるといい，該当セッションもしくはマッチングセッションが *locally exposed* である場合に，鍵共有セッションが暴露であるという．

このUMモデルにおける安全性を以下のように定義する．

定義 4. *(SK 安全性: Session Key security)[2]* セッション鍵共有プロトコル (SKS) に対する *UM* 攻撃者を A とする．A は任意のタイミングで終了，未失効，未暴露セッションの中からセッションを選び，以下の *Test* オラクルを呼び出すことができる．

Test: 選んだセッションにおいて，$b = 0$ ならば A にランダムな値を返す．$b = 1$ ならばセッション鍵 K を返す．

A は *Test* オラクル呼出し後，b' を出力し終了する．任意の *UM* 攻撃者 A について以下の条件を満たす場合に，セッション鍵共有プロトコル (SKS) は *SK* 安全であるという．

- *SKS* は，コラプトされていない *2* つのパーティのマッチングセッションが終了すると，*2* つのパーティは必ず同じセッション鍵 K を出力する．
- $\mathrm{Adv}^{sk}_{SKS,A} := \left|\Pr[b' = b] - \frac{1}{2}\right| \leq \epsilon$ が成り立つ．

2.4 デバイス鍵共有プロトコル (DKS: Device Key Share Protocol)

デバイス鍵共有プロトコル (DKS) は，以下の二つのアルゴリズムから構成される．ここで，$\mathcal{K}$ をセッション鍵空間，$\mathcal{K}_{dk}$ をデバイス鍵空間とする．

- $(C_{dk}, dk) \leftarrow \mathrm{DKS.Enc}(K)$: セッション鍵 $K \in \mathcal{K}$ を入力とし，デバイス鍵 $dk \in \mathcal{K}_{dk}$，デバイス鍵 dk の暗号文 C_{dk} を出力する．
- dk or $\perp \leftarrow \mathrm{DKS.Dec}(K, C_{dk})$: K, C_{dk} を入力とし，C_{dk} の復号結果を返す．

このとき，すべての $K \in \mathcal{K}$ に対して，$(C_{dk}, dk) \leftarrow \mathrm{DKS.Enc}(K)$ であるならば，$\mathrm{DKS.Dec}(K, C_{dk}) = dk$ を満たす．

定義 5. デバイス鍵共有プロトコルにおける *indistinguishability against chosen-ciphertext attack (IND-CCA)* は，以下のように定義される．A をデバイス鍵共有プロトコル *DKS=*{*DKS.Enc,DKS.Dec*} に対する多項式時間攻撃者とする．ここで，以下のゲームを考える．

Step 1: $K \leftarrow_R \mathcal{K}$, $S \leftarrow \emptyset$.

Step 2: $b \leftarrow_R \{0,1\}$,$b' \leftarrow A^{O_1, O_2}(1^\lambda)$.

ここで，O_1 は *DKS* の暗号化オラクルであり，オラクルを呼び出すと dk_0, dk_1 をランダムで選び，dk_b を暗号化して (C_{dk}, dk_b) を返す．O_2 は *DKS* の復号オラクルであり，C_{dk} を渡すとその復号結果を返す．それぞれ以下のように定義する．

O_1: $(C_{dk}, dk_0) \leftarrow DKS.Enc(K), dk_1 \leftarrow_R \mathcal{K}_{dk}$, $S \leftarrow S \cup \{C_{dk}\}$. *Return* (C_{dk}, dk_b)

O_2: *If* $C_{dk} \in S$, *return* $\perp$. *Else* $dk \leftarrow DKS.Dec(K, C_{dk})$, *return* dk.

このとき，上記のゲームにおける A のアドバンテージは $\mathrm{Adv}^{ind-cca}_{DKS,A} := \left|\Pr[b' = b] - \frac{1}{2}\right|$ という式で定義される．グループ鍵共有プロトコルが *IND-CCA* 安全性を満たすとは，任意の多項式攻撃者 A に対して，$\mathrm{Adv}^{ind-cca}_{DKS,A} \leq \epsilon$ となるときである．

2.5 グループ鍵共有プロトコル (GKS: Group Key Share Protocol)

グループ鍵共有プロトコル (GKS) は，以下の 2 つのアルゴリズムから構成される．ここで，$\mathcal{K}_{gk}$ をグループ鍵空間とする．

- $(C_{gk}, gk) \leftarrow \mathrm{GKS.Enc}(G, dk)$: あるユーザ集合 G とデバイス鍵 $dk \in \mathcal{K}_{dk}$ を入力とし，グループ鍵 $gk \in \mathcal{K}_{gk}$，グループ鍵 gk の暗号文 C_{gk} を出力する．
- gk or $\perp \leftarrow \mathrm{GKS.Dec}(dk, C_{gk})$: G, dk, C_{gk} を入力とし，C_{gk} の復号結果を返す．

このとき，すべての $dk \in \mathcal{K}_{dk}$ に対して，$(C_{gk}, gk) \leftarrow \mathrm{GKS.Enc}(G, dk)$ であるならば，$\mathrm{GKS.Dec}(dk, C_{gk}) = gk$ を満たす．

定義 6. グループ鍵共有プロトコルにおける *indistinguishability against chosen-ciphertext attack (IND-CCA)* は，以下のように定義される．A をグループ鍵共有プロトコル *GKS=*{*GKS.Enc,GKS.Dec*} に対する多項式時間攻撃者とする．ここで，以下のゲームを考える．

Step 1: $dk \leftarrow_R \mathcal{K}_{dk}$,$S \leftarrow \emptyset$.

Step 2: $(C^*_{gk}, gk_1) \leftarrow GKS.Enc(G, dk)$.

Step 3: $gk_0 \leftarrow_R \mathcal{K}_{gk}$,$b \leftarrow_R \{0,1\}$.

Step 4: $b' \leftarrow A^{O_1, O_2}(C^*_{gk}, gk_b)$.

ここで，O_1 は *GKS* の暗号化オラクルであり，O_1 を呼び出すと (C_{gk}, gk) を返す．O_2 は *GKS* の復号オラクルであり，C_{gk} を入力するとその復号結果を返す．それぞれ以下のように定義する．

O_1: $(C_{gk}, gk) \leftarrow GKS.Enc(dk)$, $S \leftarrow S \cup \{C_{gk}\}$, *return* (C_{gk}, gk)

O_2: *If* $C_{gk} = C^*_{gk}$ *or* $C_{gk} \in S$, *return* $\perp$. *Else* $gk \leftarrow GKS.Dec(dk, C_{gk})$, *return* gk.

このとき，上記のゲームにおける A のアドバンテージは，$\mathrm{Adv}^{ind-cca}_{GKS,A} := \left|\Pr[b' = b] - \frac{1}{2}\right|$ という式で定義される．グループ鍵共有プロトコルが *IND-CCA* 安全性を満たすとは，任意の多項式攻撃者 A に対して，$\mathrm{Adv}^{ind-cca}_{GKS,A} \leq \epsilon$ となるときである．

3. 機器認証プロトコル

3.1 モデル

機器認証 (DA: Deveice Authentication) プロトコルのモデルを定義する．モデルで用いるパラメータを次のように

設定する: 全ユーザ数を n とする．多項式時間で動く DA ユーザの集合を $\mathcal{U} := \{U_1, U_2, \ldots, U_n\}$ とする．グループ G を $\mathcal{U}$ の部分集合として, G におけるセッション ID を sid とする．また，ユーザ $U_i \in \mathcal{U}$ において, セッション ID sid で共有されたセッション鍵を $K_{U_i}^{sid}$，デバイス鍵を $dk_{U_i}^{sid}$ とする．

DA プロトコルは DKS = {DKS.Enc, DKS.Dec}, GKS = {GKS.Enc, GKS.Dec}, DEM = {DEM.Enc, DEM.Dec} を用いて次のように定義される．

- $C \leftarrow$ DA.Enc(U, G, M): DA.Enc は DKS, GKS, DEM で構成される．次のプロトコルを用いて, ユーザ $U \in G$ から $G \subset \mathcal{U}$ へメッセージ $M \in \mathcal{M}$ を送信する．
  - グループマネージャ $GM \in G$ から各ユーザ $U_i \in G$ へデバイス鍵を配布．
    - $K_{U_i} \leftarrow$ SKS(GM, U_i, sid),
    - $(C_{dk_{U_i}}, dk_{U_i}) \leftarrow$ DKS.Enc(K_{U_i}),
    - $C_{dk} = (C_{dk_{U_1}}, \ldots, C_{dk_{U_n}})$,
  - GM から全ての $U_i \in G$ へグループ鍵を配布．
    - $(C_{gk_G,U_i}, gk_G) \leftarrow$ GKS.Enc(G, dk_{U_i}),
    - $C_{gk_G} = (C_{gk_G,U_1}, \ldots, C_{gk_G,U_n})$,
  - U から $U_i \in G$ へメッセージを送信．
    - $C_M \leftarrow$ DEM.Enc(gk_G, M),
  - return $C = (C_M, C_{gk_G}, C_{dk})$.
- M or $\perp \leftarrow$ DA.Dec(G, K_{U_i}, C): GM とセッション鍵の共有を行った各ユーザ $U_i \in G$ は C を次のように復号する．
  - $C = (C_M, C_{gk_G}, C_{dk})$,
  - $dk_{U_i} \leftarrow$ DKS.Dec$(K_{U_i}, C_{dk_{U_i}})$,
  - $gk_G \leftarrow$ GKS.Dec(G, dk_{U_i}, C_{gk_G}),
  - return $M/\perp =$ DEM.Dec(gk_G, C_M).

また, 時刻区間の数を T として, グループ鍵の時刻区間のインデックスを $t \in \{1, 2, \ldots, T\}$ とする．ある時刻 t におけるグループ鍵更新を以下のアルゴリズムで構成する．

- $(C_{gk_{G,t}}, gk_{G,t}) \leftarrow$ GKUpd.Enc $(G, t, dk_{U_i}^{sid})$: 時刻インデックス $t \in [T]$ における鍵更新のために, GM はデバイス鍵を用いて新しいグループ鍵 $gk_{G,t}$ をカプセル化した暗号文を各ユーザに送る．
- $gk_{G,t}$ or $\perp \leftarrow$ GKUpd.Dec $(G, dk_{U_i}^{sid}, C_{gk_{G,t}})$: 各ユーザは, 新しいグループ鍵をカプセル化した暗号文を受け取り, 自身のデバイス鍵で復号する．

3.2 安全性定義

DA プロトコルの安全性として, IND-iCCA 安全性 (indistinguishability against insider chosen ciphertext attack) と IND-KE-CCA 安全性 (indistinguishability against key exposure and chosen ciphertext attack) を定義する．

定義 7 (内部攻撃者に対する IND-CCA 安全性 (IND-iCCA 安全性))．*DA* に対する多項式時間攻撃者を A，多項式時間で動く *DA* のユーザ集合を $\mathcal{U} := \{U_1, U_2, ..., U_n\}$ とする．

(1) $b \leftarrow_R \{0,1\}$, $S_G \leftarrow \emptyset$ とする．A は以下のオラクルにアクセスできる．

- *Initialize*(G)*:* 入力として集合 $G \subset \mathcal{U}$ を受け取り, ユーザ $U_i \in G$ を初期化し，そのセッション *ID* sid を出力する．
- *Invoke*(G', M)*:* 入力として集合 G', メッセージ $M \in \mathcal{M}$ を受け取る．
- *Corrupt*(G)*:* 入力として集合 $G \subset \mathcal{U}$ を受け取り, G で生成されたグループ鍵の集合 $\{gk_{G,t}\}_{t\in[T]}$ を出力する． ここで, G が *Corrupt* オラクルに対して発行された場合, G はコラプトされていると記述する．

(2) A は，任意のタイミングでコラプトされていない $G_{tg} \subset \mathcal{U}$ をターゲットグループとして選び，以下のオラクルにクエリを発行することができる．ただし, *(1)* のオラクルで発行したグループ G, または G の部分集合を選ぶことはできない．

- *TestLR*(G_{tg}, M_0, M_1)*:* 入力として G_{tg} と *2* つのメッセージ $M_0, M_1 \in \mathcal{M}$ を受け取り, $C \leftarrow$ DA.Enc$(\cdot, G_{tg}, M_b)$ を出力する．このとき, $S_G \leftarrow S_G \cup \{C\}$ とする．
- *Dec*(G_{tg}, C)*:* 入力として G_{tg} と暗号文 C を受け取り, G_{tg} のグループ鍵 $gk_{G_{tg}}$ で復号した結果を出力する．このとき, $C \in S_G$ ならば $\perp$ を返す．

(3) A は, *(1)* のオラクルにクエリを発行する．ただし, 集合 $G \subseteq G_{tg}$ は発行できない．

(4) A は，テストクエリを含むクエリを多項式回発行したのちに $b' \in \{0,1\}$ を出力する．

このとき，上記のゲームにおける A アドバンテージは，$\mathrm{Adv}_{DA,A}^{ind-icca} := \left|\Pr[b' = b] - \frac{1}{2}\right|$ という式で表される．機器認証プロトコルが，内部攻撃者に対して *IND-iCCA* 安全性を満たすとは，上記の多項式時間攻撃者 A に対して，$\mathrm{Adv}_{DA,A}^{ind-icca} \leq \epsilon$ となるときである．

定義 8 (鍵漏洩に対する IND-CCA 安全性 (IND-KE-CCA 安全性))．*DA* に対する多項式時間攻撃者を A，多項式時間で動く *DA* のユーザ集合を $\mathcal{U} := \{U_1, U_2, ..., U_n\}$ とする．

(1) $\boldsymbol{b} = (b_1, \ldots, b_T) \leftarrow_R \{0,1\}^T$, リスト $S_G \leftarrow \emptyset$, リスト $S_{exp} \leftarrow \emptyset$ とする．A は以下のオラクルにアクセスできる．

- *Expose*(t)*:* 入力として時刻インデックス $t \in [T]$ を受け取り, $gk_{G,t}$ を出力する． $S_{exp} \leftarrow S_{exp} \cup \{t\}$ とする．
- *Update*(t)*:* 入力として時刻インデックス $t \in [T]$ を受け取り, $C_{gk_{G,t}}$ を出力する．
- *LR*(t, M_0, M_1)*:* 入力として時刻インデックス t と *2* つのメッセージ $M_0, M_1 \in \mathcal{M}$ を受け取り, $C \leftarrow$ DA.Enc(U, G, M_b) を出力する．このとき, *GKS.Enc* アルゴリズムの代わりに *GKUpd.Enc* アルゴリズム

を実行する．また，$S_G \leftarrow S_G \cup \{(t, C)\}$ とする．

- *Decrypt*(t, C, U_i): 入力として時刻インデックス t と暗号文 C，ユーザ $U_i \in G$ を受け取り，$M/\perp = \mathrm{DA.Dec}(G, K_{U_i}, C)$ を出力する．このとき，$(t, C) \in S_G$ ならば $\perp$ を出力する．

(2) A は，それぞれのオラクルに多項式回クエリを発行したのちに $(t, b') \in [T] \times \{0, 1\}$ を出力する．

このとき，上記のゲームにおける A アドバンテージは，$\mathrm{Adv}^{ind-ke-cca}_{DA,A} := \left|\Pr[(t \notin S_{exp}) \wedge (b' = b_t)] - \frac{1}{2}\right|$ という式で表される．機器認証プロトコルが，*IND-KE-CCA* 安全性を満たすとは，上記の多項式時間攻撃者 A に対して，$\mathrm{Adv}^{ind-ke-cca}_{DA,A} \leq \epsilon$ となるときである．

3.3 構成

鍵更新可能な DA プロトコル DA = {DA.Enc, DA.Dec, GKUpd.Enc, GKUpd.Dec} は，SKS, DKS = {DKS.Enc, DKS.Dec}, AE = {AE.Enc, AE.Dec} を用いて次のように構成される．3.1 節との違いは，GKS と DEM, GKUpd では具体的に AE が用いられる．

- $C \leftarrow \mathrm{DA.Enc}(U, G, M)$:
  - グループマネージャ $GM \in G$ から各ユーザ $U_i \in G$ へデバイス鍵を配布．
    - $K_{U_i} \leftarrow \mathrm{SKS}(GM, U_i, sid)$,
    - $(C_{dk_{U_i}}, dk_{U_i}) \leftarrow \mathrm{DKS.Enc}(K_{U_i})$,
    - $C_{dk} = (C_{dk_{U_1}}, \ldots, C_{dk_{U_n}})$,
  - GM から全ての $U_i \in G$ へグループ鍵を配布．
    - $gk_G \leftarrow_R \mathcal{K}_{gk}$,
    - $C_{gk_G, U_i} \leftarrow \mathrm{AE.Enc}(dk_{U_i}, gk_G)$,
    - $C_{gk_G} = (C_{gk_G, U_1}, \ldots, C_{gk_G, U_n})$,
  - U から $U_i \in G$ へメッセージを送信．
    - $C_M \leftarrow \mathrm{AE.Enc}(gk_G, M)$,
  - return $C = (C_M, C_{gk_G}, C_{dk})$.
- M or $\perp \leftarrow \mathrm{DA.Dec}(G, K_{U_i}, C)$:
  - $C = (C_M, C_{gk}, C_{dk_{U_i}})$,
  - $dk_{U_i} \leftarrow \mathrm{DKS.Dec}(K, C_{dk_{U_i}})$,
  - $gk_G \leftarrow \mathrm{AE.Dec}(G, dk_{U_i}, C_{gk})$,
  - return $M/\perp = \mathrm{AE.Dec}(gk_{G,t}, C_M)$.
- $(C_{gk_{G,t}}, gk_{G,t}) \leftarrow \mathrm{GKUpd.Enc}\ (G, t, dk_{U_i})$:
  - $gk_{G,t} \leftarrow_R \mathcal{K}_{gk}$,
  - $C_{gk_{G,t}} \leftarrow \mathrm{AE.Enc}(dk, gk_{G,t})$,
  - return $(C_{gk_{G,t}}, gk_{G,t})$.
- $gk_{G,t}$ or $\perp \leftarrow \mathrm{GKUpd.Dec}\ (G, dk_{U_i}, C_{gk_{G,t}})$:
  - return $M/\perp = \mathrm{AE.Dec}(dk, C_{gk_{G,t}})$.

4. 安全性証明

提案構成法が内部攻撃者に IND-CCA 安全性を満たし鍵漏洩に対しても安全性を達成することを示す．

4.1 内部攻撃者に対する安全性

定理 9. *SKS* が *SK* 安全，*DKS* が *IND-CCA* 安全，*AE* が *IND-CCA* 安全ならば，*DA* は *IND-iCCA* 安全性を満たす．

証明: A を DA に対する多項式時間内部攻撃者とする．この時，以下のゲームを考える．また，X_i を Game i において $b' = b$ となる事象とする．

Game 0: DA の IND-iCCA ゲームである．したがって，以下の式を得る．

$$\mathrm{Adv}^{ind-icca}_{DA,A} = \left|\Pr[X_0] - \frac{1}{2}\right| \tag{1}$$

Game 1: Game 0 の DA アルゴリズムにおいて DKS の入力に用いられる K_{U_i} の値をランダムなものに変える．ここで，SKS に対する多項式時間攻撃者を A_{sks} とし，以下のゲームを構成する．

Step 1:

- $\beta \leftarrow_R \{0, 1\}$

Step 2:

- $b \leftarrow_R \{0, 1\}$.
- A がクエリ G を Initialize オラクルに入力したら，A_{sks} は G 内の各ユーザ U_i を SKS の Initialize オラクルを入力し，返答を sid とする．また，$S_G \leftarrow \emptyset$ とする．
- A がクエリ (G', M) を Invoke オラクルに入力したら，$gk_{G'} \leftarrow_R \mathcal{K}_{gk}$ とし A_{sks} は以下のように動作する．
  (1) グループマネージャ $GM \in G'$ を設定し，GM 以外の各ユーザを $U_i \in G'$ とする．GM と各 U_i に対してセッション sid を SKS の Invoke オラクルに入力する．
  (2) GM と各 U_i に対してを SKS の Test クエリを発行し，返答を K_{U_i} に割り当てる．
  (3) 受信者を U_i とするグループ G' 内の各セッションについて
  $(C_{dk_{U_i}}, dk_{U_i}) \leftarrow \mathrm{DKS.Enc}(K_{U_i})$,
  $C_{gk'_G, U_i} \leftarrow \mathrm{AE.Enc}(dk_{U_i}, gk_{G'})$,
  $C_M \leftarrow \mathrm{AE.Enc}(gk_{G'}, M)$ とし，
  $C_{dk} := (C_{dk_{U_1}}, \ldots, C_{dk_{U_i}})$,
  $C_{gk_G} := (C_{gk_G, U_1}, \ldots, C_{gk_G, U_i})$,
  $C := (C_M, C_{dk}, C_{gk'_G})$ とする．
- A がクエリ G を Corrupt オラクルに入力したら，A_{sks} はグループ G を失効させ gk_G を返す．
- A がクエリ (G_{tg}, M_0, M_1) を TestLR オラクルに入力したら，A_{sks} は以下のように動作する．
  (1) $C_{M_b} \leftarrow \mathrm{AE.Enc}(gk_{G_{tg}}, M_b)$,
  (2) G_{tg} における暗号文 $C = (C_{M_b}, C_{dk}, C_{gk_{G_{tg}}})$ を返し，$S_G \leftarrow S_G \cup \{C\}$ とする．
- A がクエリ (G_{tg}, C) を Dec オラクルに入力したら，A_{sks} は G_{tg} 内のユーザ $U_i \in G_{tg}$ をランダムで選び，以下のように動作する．
  (1) if $C \in S_{G_{tg}}$, return $\perp$.

(2) else $C = (C_M, C_{dk}, C_{gk_{G_{tg}}})$.
(3) $dk_{U_i} \leftarrow \mathrm{DKS.Dec}(K_{U_i}, C_{dk_{U_i}})$,
(4) $gk_{G_{tg}} \leftarrow \mathrm{AE.Dec}(dk_{U_i}, C_{gk_{G_{tg}},U_i})$
(5) $M \leftarrow \mathrm{AE.Dec}(gk_{G_{tg}}, C_M)$，$M$ を返す．

- A が b' を出力して停止したら A_{sks} は以下の動作を行い停止する．
  - if $b' = b$, return $\beta' = 1$.
  - else return $\beta' = 0$.

この時，A_{sks} はオラクルを正しくシミュレートしているので，以下の式を得る．

$$\left|\Pr[X_0] - \Pr[X_1]\right| \leq \mathrm{Adv}^{sk}_{SKS,A} \tag{2}$$

Game 2:Game 1 のユーザ U_i の持つ各セッションのデバイス鍵 dk_{U_i} をランダムな値に変更する．ここで，DKS に対する多項式時間攻撃者を A_{dks} とし，以下のゲームを構成する．

Step 1:

- $\beta \leftarrow_R \{0,1\}$

Step 2:

- $b \leftarrow_R \{0,1\}$
- A がクエリ G を Initialize オラクルに入力したら，A_{dks} は G 内のユーザすべてを初期化し sid を返す．また，$S_G \leftarrow \emptyset$ とする．
- A がクエリ (G', M) を Invoke オラクルに入力したら，$gk_{G'} \leftarrow_R \mathcal{K}_{gk}$ とし A_{sks} は以下のように動作する．
  (1) グループマネージャ $GM \in G'$ を設定し, GM 以外の各ユーザを $U_i \in G'$ とする．GM と各 U_i のセッション鍵 K_{U_i} をランダムに選ぶ.
  (2) DKS のオラクル O_1 を呼び出し，出力を $C_{dk_{U_i}}, dk_{U_i}$ とする．
  (3) $C_{gk_{G'},U_i} \leftarrow \mathrm{AE.Enc}(dk_{U_i}, gk_{G'})$,
  (4) $C_M \leftarrow \mathrm{AE.Enc}(gk_G, M)$,
  (5) $C_{dk} := (C_{dk_{U_1}}, \ldots, C_{dk_{U_i}})$,
  $C_{gk_G} := (C_{gk_G,U_1}, \ldots, C_{gk_G,U_i})$,
  $C := (C_M, C_{dk}, C_{gk'_G})$ とする．
- A がクエリ G を Corrupt オラクルに入力したら，A_{dks} はグループ G を失効させ gk_G を返す．
- A がクエリ (G_{tg}, M_0, M_1) を TestLR オラクルに入力したら，A_{dks} は以下のように動作する．
  (1) $C_{M_b} \leftarrow \mathrm{AE.Enc}(gk_{G_{tg}}, M_b)$.
  (2) G_{tg} における暗号文 $C = (C_{M_b}, C_{dk}, C_{gk_{G_{tg}}})$ を返し，$S_{G_{tg}} \leftarrow S_{G_{tg}} \cup C$ とする．
- A がクエリ (G_{tg}, C) を Dec オラクルに入力したら，A_{sks} は G_{tg} 内のユーザ $U_i \in G_{tg}$ をランダムで選び，以下のように動作する．
  (1) if $C \in S_{G_{tg}}$, return $\perp$.
  (2) else $C = (C_M, C_{dk_{U_i}}, C_{gk_{G_{tg}}})$,
  (3) $dk_{U_i} \leftarrow \mathrm{DKS.Dec}(K_{U_i}, C_{dk_{U_i}})$,
  (4) $gk_{G_{tg}} \leftarrow \mathrm{AE.Dec}(dk_{U_i}, C_{gk_{G_{tg}},U_i})$,
  (5) $M \leftarrow \mathrm{AE.Dec}(gk_{G_{tg}}, C_M)$，$M$ を返す．
- A が b' を出力して停止したら A_{dks} は以下の動作を行い停止する．
  - if $b' = b$, return $\beta' = 1$.
  - else return $\beta' = 0$.

この時，A_{dks} はオラクルを正しくシミュレートしているので，以下の式を得る．

$$\left|\Pr[X_1] - \Pr[X_2]\right| \leq \mathrm{Adv}^{ind-cca}_{DKS,A_{dks}} \tag{3}$$

Game 3:Game 2 の各グループで用いられるグループ鍵 gk_G をランダムな値に変更する．ここで，AE に対する多項式時間攻撃者を A_{ae} とし，以下のゲームを構成する．

Step 1:

- $\beta \leftarrow_R \{0,1\}$

Step 2:

- $b \leftarrow_R \{0,1\}$
- A がクエリ G を Initialize オラクルに入力したら，A_{ae} は G 内のすべてのユーザを初期化する．また，$S_G \leftarrow \emptyset$ とする．
- A がクエリ (G', M) を Invoke オラクルに入力したら，$gk_{0G'}, gk_{1G'} \leftarrow_R gk$ とし A_{ae} は以下のように動作する．
  (1) グループマネージャ $GM \in G'$ を設定し，GM 以外の各ユーザを $U_i \in G'$ とする．GM と各 U_i に対して，$K_{U_i}, C_{dk_{U_i}}, dk_{U_i}$ をランダムに選ぶ.
  (2) AE のオラクル O_1 に (gk_{0G}, gk_{1G}) を入力し，出力を C_{gk_G,U_i} とする．
  (3) $C_M \leftarrow \mathrm{AE.Enc}(gk_{0G'}, M)$,
  (4) $C_{dk} := (C_{dk_{U_1}}, \ldots, C_{dk_{U_i}})$,
  $C_{gk_G} := (C_{gk_G,U_1}, \ldots, C_{gk_G,U_i})$,
  $C := (C_M, C_{dk}, C_{gk'_G})$ とする．
- A がクエリ G を Corrupt オラクルに入力したら，A_{sks} はグループ G を失効させ gk_G を返す．
- A がクエリ (G_{tg}, M_0, M_1) を TestLR オラクルに入力したら，A_{ae} は以下のように動作する．
  (1) $C_{M_b} \leftarrow \mathrm{AE.Enc}(gk_{0G_{tg}}, M_b)$.
  (2) G 内の各セッションにおける暗号文 $C = (C_{M_b}, C_{dk}, C_{gk_{G_{tg}}})$ を返し，$S_G \leftarrow S_G \cup C$ とする．
- A がクエリ (G_{tg}, C) を Dec オラクルに入力したら，A_{sks} は G_{tg} 内のユーザ $U_i \in G_{tg}$ をランダムで選び，以下のように動作する．
  (1) if $C \in S_{G_{tg}}$, return $\perp$.
  (2) else $C = (C_M, C_{dk_{U_i}}, C_{gkG_{tg}})$, AE のオラクル O_2 に $C_{gk_{G_{tg}}}$ を入力し，返答を $gk_{G_{tg}}$ とする．
  (3) $M \leftarrow \mathrm{AE.Dec}(gk_{G_{tg}}, C_M)$，$M$ を返す．
- A が b' を出力して停止したら A_{ae} は以下の動作を行い停止する．

– if $b' = b$, return $\beta' = 1$.
– else return $\beta' = 0$.

この時，A_{sks} はオラクルを正しくシミュレートしているので，以下の式を得る．

$$\left|\Pr[X_2] - \Pr[X_3]\right| \leq \mathrm{Adv}^{ind-cca}_{AE,A_{ae}} \quad (4)$$

またこの時，もう一度 AE に対する多項式時間攻撃者 A_{ae} を考え，以下のゲームを構成する．

Step 1:

- $b \leftarrow_R \{0,1\}$

Step 2:

- A がクエリ G を Initialize オラクルに入力したら，A_{ae} は G 内のすべてのユーザを初期化する．また，$S_G \leftarrow \emptyset$ とする．
- A がクエリ (G', M) を Invoke オラクルに入力したら，A_{ae} は以下のように動作する．
  (1) グループマネージャ $GM \in G'$ を設定し，GM 以外の各ユーザを U_i とする． GM と各 U_i に対して，$K_{U_i}, C_{dk}, dk_{U_i}, C_{gk'_G}, gk_{G',t}$ をランダムに選ぶ．
  (2) M, M を AE のオラクル O_1 に入力する．返答を C_M とする．
  (3) $C_{dk} := (C_{dk_{U_1}}, \ldots, C_{dk_{U_i}})$,
  $C_{gk_G} := (C_{gk_G,U_1}, \ldots, C_{gk_G,U_i})$,
  $C := (C_M, C_{dk}, C_{gk'_G})$ とする．
- A がクエリ G を Corrupt オラクルに入力したら，A_{sks} はグループ G を失効させ gk_G を返す．
- A がクエリ (G_{tg}, M_0, M_1) を TestLR オラクルに入力したら，A_{ae} は以下のように動作する．
  (1) M_0, M_1 を AE のオラクル O_1 に入力し，返答を C_{M_b} とする．
  (2) G_{tg} 内の各セッションにおける暗号文 $C = (C_{dk}, C_{gk_{G_{tg}}}, C_{M_b})$ を返し，$S_{G_{tg}} \leftarrow S_{G_{tg}} \cup \emptyset$ とする．
- A がクエリ (G_{tg}, C) を Dec オラクルに入力したら，A_{sks} は以下のように動作する．
  (1) if $C \in S_{G_{tg}}$, return $\perp$.
  (2) else $C = (C_M, C_{dk}, C_{gk_{G_{tg}}})$, AE のオラクル O_2 に C_M を入力し，返答を M として返す．
- A が b' を出力して停止したら A_{ae} も b' を出力し停止する．

攻撃者 A が $b' = b$ となる b' を出力する時，同時に攻撃者 A_{ae} も AE の IND-CCA ゲームに対して $b' = b$ となる b' を出力することになる．つまり，この Game 3 における勝利確率は AE の安全性に依存しているので，以下の式を得ることができる．

$$\left|\Pr[X_3] - \frac{1}{2}\right| \leq \mathrm{Adv}^{ind-cca}_{AE,A_{ae}} \quad (5)$$

以上から，証明は完了した． □

4.2 鍵漏洩に対する安全性

定理 10. *SKS が SK 安全，DKS が IND-CCA 安全，AE が IND-CCA 安全ならば，DA は IND-KE-CCA 安全性を満たす．*

証明: A を DA に対する多項式時間攻撃者とする．このとき，以下のゲームを考え，X_i を Game i において，$t \in S_{exp}$ かつ $b' = b$ となる事象とする．

Game 0: DA の IND-KE-CCA ゲームである．したがって，以下の式を得る．

$$\mathrm{Adv}^{ind-ke-cca}_{DA,A} = \left|\Pr[X_0] - \frac{1}{2}\right| \quad (6)$$

Game 1: Game 0 においてセッション鍵とデバイス鍵 dk_{U_i} の値をランダムな値に変える．このとき，定理 9 の Game 2 までの証明と同様にして以下が成り立つ．

$$|\Pr[X_0] - \Pr[X_1]| \leq Adv^{sk}_{SKS} + Adv^{ind-cca}_{DKS}$$

Game 2:Game 1 においてグループ鍵 $gk_{G,t}$ をランダムな値に変更する．ここで，AE に対する多項式時間攻撃者を A_{ae1} とし，以下のゲームを構成する．

Step 1:

- $\beta \leftarrow_R \{0,1\}$,

Step 2:

- $\boldsymbol{b} = (b_1, \ldots, b_T) \leftarrow_R \{0,1\}^T$, リスト $S_{exp} \leftarrow \emptyset$, $S_{upd} \leftarrow \emptyset$, $S_G \leftarrow \emptyset$ とする．
- 全ての $U_i \in G$ と全ての $t \in [T]$ に対して，それぞれセッション鍵 K_{U_i} とグループ鍵 $gk_{G,t}$ をランダムに選ぶ．$U_i^* \in G$ を 1 つランダムに選び，全ての $U_i \in G \backslash \{U_i^*\}$ に対してデバイス鍵 dk_{U_i} をランダムに選ぶ．
- A が $t \in [T]$ を Expose オラクルに入力したら，$gk_{G,t} \leftarrow_R \mathcal{K}_{gk}$ を返し，$S_{exp} \leftarrow S_{exp} \cup \{t\}$ とする．
- A が $t \in [T]$ を Update オラクルに入力したら，U^* においては $gk'_{G,t} \leftarrow_R \mathcal{K}_{gk}$ として，$gk_{G,t}, gk'_{G,t}$ を AE のオラクル O_1 に入力し，その返答を $C_{gk_{G,t},U^*}$ とする．任意の $U_i \in G \backslash \{U_i^*\}$ において dk_{U_i} を用いて $C_{gk_{G,t},U_i}$ を計算して A に C_{gk_t} を返す． $S_{upd} \leftarrow S_{upd} \cup \{t\}$ とする．
- A が t, M_0, M_1 を LR オラクルに入力したら，次のいずれかの条件に当てはまる場合
  – $t \in S_{exp}$ のとき，$\perp$ を返す，
  – $t \in S_{upd}$ のとき，$gk_{G,t}$ を用いて $C = (C_{dk}, C_{gk_G,t}, C_{M_{b_t}})$ を計算して返し，$S_G \leftarrow S_G \cup \{(t, C)\}$ とする，
  – $t \notin S_{exp}$ かつ $t \notin S_{upd}$ のとき，U_i^* に対しては $gk'_{G,t} \leftarrow_R \mathcal{K}_{gk}$ として，$gk_{G,t}, gk'_{G,t}$ を，AE のオラクル O_1 に入力し，$C_{gk_{G,t},U_i^*}$ を受け取る．それ以外の

U_i に対しては GKUpd アルゴリズムに従って計算して，$C = (C_{dk}, C_{M_{b_t}} = AE.Enc(gk_{G,t}, M_{b_t}))$ を返し, $S_G \leftarrow S_G \cup \{(t, C)\}$ とする.

- A が t, C を Dec オラクルに入力したら, 次の処理を行う.
  (1) $(t, C) \in S_G$ ならば, $\perp$ を返す,
  (2) そうでなければ,K_{U_i} を用いて，その返答 $gk_{G,t}$ を受け取り, $M/\perp = \mathrm{AE.Dec}(gk_{G,t}, C_M)$ を返す.
- A が (t, b') を出力したとき, $t \notin S_{exp}$ かつ $t \in S_{upd}$ かつ $b' = b_t$ ならば A_{ae} は $\beta' = 1$ を出力する. そうでなければ, $\beta' = 0$ を出力する.

A が $t \notin S_{exp}$ かつ $t \in S_{upd}$ かつ $b' = b_t$ となる b' を出力していれば, A_{ae1} は AE の IND-CCA ゲームに勝利しており,$N = |G|$ とすると，以下の式を得る.

$$|\Pr[X_1] - \Pr[X_2]| \leq N \cdot Adv^{ind-cca}_{AE,A_{ae1}}$$

また，AE に対する多項式時間攻撃者を A_{ae2} とし，以下のゲームを構成する.

Step 1:

- $\beta \leftarrow_R \{0, 1\}$

Step 2:

- $\boldsymbol{b} = (b_1, \ldots, b_T) \leftarrow_R \{0,1\}^T$, リスト $S_{exp} \leftarrow \emptyset$, $S_{upd} \leftarrow \emptyset$, $S_G \leftarrow \emptyset$ とする.
- 全ての $U_i \in G$ に対してセッション鍵 K_{U_i}, デバイス鍵 dk_{U_i} をランダムに選ぶ. また, $t^* \in [T]$ を一様ランダムに 1 つ選ぶ. 全ての $t \in [T] \backslash \{t^*\}$ に対してグループ鍵 $gk_{G,t}$ をランダムに選ぶ.
- A が $t \in [T]$ を Expose オラクルに入力したら, $gk_{G,t} \leftarrow_R \mathcal{K}_{gk}$ を返し, $S_{exp} \leftarrow S_{exp} \cup \{t\}$ とする.
- A が $t \in [T]$ を Update オラクルに入力したら, 全ての $U_i \in G$ において, $C_{gk_{G,t},U_i} = AE.Enc(dk_{U_i}, gk_{G,t})$ を計算して, $C_{gk_{G,t}} = (C_{gk_{G,t},U_1}, \ldots, C_{gk_{G,t},U_N})$ を返す.
- A が t, M_0, M_1 を LR オラクルに入力したら, 次のいずれかの条件に当てはまる場合の処理を行う.
  - $t \in S_{exp}$ のとき, $\perp$ を返す,
  - $t \neq t^*$, $gk_{G,t}$ を用いて $C = (C_{dk}, C_{gk_G,t}, C_{M_b})$ を計算して返し, $S_G \leftarrow S_G \cup \{(t, C)\}$ とする,
  - $t = t^*$ のとき，C_{dk} と $C_{gk'_{G,t^*}}$ を計算し，M_0, M_1 を AE のオラクル O_1 に発行し, C_{M_β} を受け取る.
- A が t, C を Dec オラクルに入力したら, 次の処理を行う.
  (1) $(t, C) \in S_G$ ならば, $\perp$ を返す,
  (2) そうでなければ, C_M を AE のオラクル O_2 に入力し, その返答を A に返す.
- A が (t, b') を出力したら, $t = t^*$ ならば A_{ae2} も b' を出力する.

A が $t^* \notin S_{exp}$ かつ $t^* \notin S_{upd}$ かつ $b' = b^*_t$ となる b' を出力していれば, t^* において A_{ae2} は AE の IND-CCA ゲームに勝利しており, 以下の式を得る.

$$\left|\Pr[X_2] - \frac{1}{2}\right| \leq T \cdot Adv^{ind-cca}_{AE,A_{ae2}}$$

以上から, 証明が完了した. □

5. まとめ

本論文では, 増田・小椋 [9] によって提案され, 現在, エコーネットコンソーシアムにおいて検討されている機器認証プロトコルの安全性証明を行った. 著者らは, SCIS2018[8] において, エコーネットコンソーシアムにおいて検討されている機器認証プロトコルの主要部が, 外部攻撃者に対して安全であることを示していたが, 本稿では, さらに, 機器認証プロトコルの主要部以外のグループ鍵更新に関しても安全性評価を行い, また, 他グループの通信内容やグループ鍵を取得できるような内部攻撃者に対する安全性も示した. 特に, KEM/DEM の理論的枠組みにより, 機器認証プロトコルの安全性を示している点が特長的と考える.

参考文献

[1] V. Shoup,“A Proposal for an ISO Standard for Public Key Encryption(v.2.1),” ISO/IEC JTC1/SC27, N2563, http://shoup.net/papers/，2001.
[2] R. Canetti, H. Krawczyk, “Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels”, EUROCRYPTO 2001, 2001.
[3] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz, “Extensible Authentication Protocol(EAP)”, RFC3748, IANA, 2004.
[4] D. Simon, B. Aboba, and R. Hurst,“The EAP-TLS Authentication Protocol,” RFC5216，2008.
[5] Y. Hanatani, N. Ogura, Y. Ohba, L. Chen, S. Das, “A Secure Multicast Group Management and Key Distribution in IEEE 802.21, ” Security Standardisation Research Springer Internationnal Publishing. 2016.
[6] D. Forsberg, Y. Ohba, B. Patil, H. Tschofenig, and A. E. Yegin, “Protocol for Carrying Authentication for Network Access (PANA),” RFC5191, IANA, 2008.
[7] IEEE, “IEEE Standard for Local and metropolitan area networks - Part 21: Media Independent Services Framework”, IEEE Std 802.21 2017, 2017.
[8] 四方順司, 打越忠宏, 海老名将宏, 佐藤慎悟, 増田洋一, 海上勇二, 高添智樹, “HEMS における機器認証プロトコルの安全性証明 ”, 2018 Symposium on Cryptography and Information Security (SCIS 2018), Niigata, Japan, Jan. 23 - 26, 2018.
[9] 増田洋一, 小椋直樹, “HEMS における機器認証とグループ鍵管理 ”, 2018 Symposium on Cryptography and Information Security (SCIS 2018), Niigata, Japan, Jan. 23 - 26, 2018.
[10] 村上隆史, “ECHONET Lite を取り巻く状況, ” pp. 16-20, 照明学会，2016.