Computer Security Symposium 2017
23 - 25 October 2017

ハッシュ連鎖アグリゲーションの応用

栗原　勇太[1,a)]　双紙　正和[1,b)]

概要：IoT (Internet of things) の普及に伴い，計算能力やメモリ，バッテリ等に制約のあるデバイスをネットワーク接続する機会が増加しており，ハッシュ関数を利用した軽量かつ効率の良いプロトコルの需要が高まっている．そこで，新しいハッシュ連鎖構成法 *hash chain aggregation* (HCA)，及び，それを用いた，二者間での共通鍵生成法が提案された．*hash chain aggregation* (HCA) は，ハッシュ関数のみを暗号プリミティブとして用いており効率が良いが，さらなる応用と，セキュリティに対する詳細な評価が課題であった．本論文では，*hash chain aggregation* (HCA) のさらなる応用について議論するとともに，より詳細なセキュリティ評価を行う．

Application of Hash Chain Aggregation

Yuta Kurihara[1,a)]　Masakazu Soshi[1,b)]

Abstract: 'IoT' (Internet of Things) environments interconnect many IoT devices, many of which often suffer from limited computational resources. Thus in order to provide lightweight and efficient authentication protocols in such environments, we have proposed a novel hash chain construction, hash chain aggregation (HCA). In this paper we discuss security and application of HCA in a detailed manner.

1. 概要

近年，Internet of Things (IoT) [1] と呼ばれるネットワーク環境が注目されるなど，計算能力やメモリ，バッテリ等に制約のあるデバイスを相互接続する機会が増加しており，ハッシュ関数などを用いた軽量かつ効率の良いプロトコルの需要が高まっている．

ハッシュ関数とは，一方向性と衝突困難性をもつ関数 (暗号プリミティブ) [2] である．また，ハッシュ関数は，量子計算機による攻撃にも耐えうると信じられており [3]，近年再び脚光を浴びている．一方で, 量子コンピュータが効率的に素因数分解問題や離散対数問題を解いてしまうことは良く知られている [4]，現代の多くの暗号システムはそれらの困難性に基づいており，量子コンピュータを用いれば徐々に突破されていくであろう．

このようなハッシュ関数の応用で重要なものに，ハッシュ連鎖がある．ハッシュ連鎖とは，入力にハッシュ関数を繰り返し適用して得られるハッシュ値の列であり，様々な暗号プロトコルの要素技術として用いられる [5], [6], [7], [8]．しかしながら，それらのプロトコルにおけるハッシュ連鎖の使用は，単純なものにとどまっており，ハッシュ連鎖のポテンシャルを生かしきれていない．また，これまでハッシュ関数のみを用いた相互認証プロトコルは知られていない．

そこで，全く新しく，かつ柔軟なハッシュ連鎖構成法 “ハッシュ連鎖アグリゲーション” が提案された．また，ハッシュ連鎖アグリゲーションを用いた，二者間での共通鍵生成法も提案された．. ハッシュ連鎖アグリゲーションを用いた認証は，以下のような重要な利点を持つ: (1) ハッシュ関数のみを暗号プリミティブとして用いており，効率がよい．(2) 全く新しいハッシュ連鎖構成法 (ハッシュ連鎖アグリゲーション) に基づいている．(3) 相手の ID のみを知っていればよく，鍵生成に通信を必要としない，ID-based 認証プロトコルである．

本論文では，この手法のセキュリティについてさらに詳

[1] 広島市立大学
Hiroshima City University
[a)] kurihara@sos.info.hiroshima-cu.ac.jp
[b)] soshi@hiroshima-cu.ac.jp

© 2017 Information Processing Society of Japan

細に考察する．また，ハッシュ連鎖アグリゲーションのさらなる応用について議論し，ハッシュ連鎖アグリゲーションの有用性をより高める．

本論文は以下のように構成される．

2 節では，ハッシュ連鎖アグリゲーション，及びそれを用いた共通鍵作成法について説明し，3 節では，ハッシュ連鎖アグリゲーションの，より詳細なセキュリティ評価を行う．4 節では，ハッシュ連鎖アグリゲーションのさらなる応用について議論する．最後に，5 にて結論を述べる．

2. ハッシュ連鎖アグリゲーション

本節では，ハッシュ連鎖アグリゲーション，及びそれを用いた共通鍵作成法 [13] について説明する．

まず，ハッシュ連鎖を定義する．ハッシュ連鎖（hash chain) とは，種 s にハッシュ関数 h を繰り返し適用して得られるハッシュ値の列である．ここで，$h^1(s) := h(s)$，$i \geq 2$ について，$h^i(s) := h(h^{i-1}(s))$ と定義する．このとき，長さ n のハッシュ連鎖は，集合 $\{1, 2, ..., n\}$ の一つの置換 $(i_1, i_2, ..., i_n)$ と，種 s を用いて，

$$(v_1, v_2, ..., v_n) \quad \text{について } v_j = h^{i_j}(s), 1 \leq j \leq n \quad (1)$$

と定義される．ここで，n 個のハッシュ値は，$h(s), h^2(s), \ldots, h^n(s)$ の順で計算されることに注意せよ．本論文では，それらのハッシュ値を，任意の順で並べ替えたものを，ハッシュ連鎖と呼ぶ．

このプロトコルにおけるプレイヤーは，Key Generation Center (KGC) と，N 人のユーザである．KGC は信頼できるものとし，KGC とそれぞれのユーザの間には，安全なチャンネルがあるとする

2.1 基本的なハッシュ連鎖

以降では，記述を簡単にするため，ある正整数 m について，$N = 2^m$ と仮定する．また，ある系列 $Q = (q_1, q_2, \ldots, q_j)$ について，i 番目の要素 q_i $(1 \leq i \leq j)$ を $Q[i]$ と書くことにする．

ハッシュ連鎖構成法を定義するために，まずタイプ I からタイプ IV までのハッシュ連鎖を定義する．以降では，ある正整数 b について，ハッシュ連鎖の長さを $\ell = 2^b$ $(1 \leq b \leq m)$ と仮定し，それぞれのハッシュ連鎖を式 (1) のような列として考える．また以下の定義では，$1 \leq i \leq \ell$ とし，s をハッシュ連鎖の seed とする．また，以降では，ハッシュ関数を h とする．

- タイプ I ハッシュ連鎖

$$C_\ell^I(s) := (v_1, v_2, \ldots, v_\ell) \quad \text{where } v_i = h^i(s)$$

- タイプ II ハッシュ連鎖

$$C_\ell^{II}(s) := (v_1, v_2, \ldots, v_\ell) \quad \text{where } v_i = h^{\ell-i+1}(s)$$

- タイプ III ハッシュ連鎖

$$C_\ell^{III}(s) := (v_1, v_2, \ldots, v_\ell)$$

$$\text{where } v_i = h^{((i-\ell/2-1) \bmod \ell)+1}(s)$$

- タイプ IV ハッシュ連鎖

$$C_\ell^{IV}(s) := (v_1, v_2, \ldots, v_\ell)$$

$$\text{where } v_i = h^{((\ell/2-i) \bmod \ell)+1}(s)$$

2.2 ハッシュ連鎖リスト

2.1 節の定義を用いれば，ハッシュ連鎖リスト (HCL) $\mathcal{L}(\tau, k, s_1, ..., s_k)$ を定義することが出来る．

$$\mathcal{L}(\tau, k, s_1, ..., s_k) := (C_{N/k}^\tau(s_1), C_{N/k}^\tau(s_2), ..., C_{N/k}^\tau(s_k)) \quad (2)$$

ここで，ある正整数 u (ただし $0 \leq u < m$) が存在して，$k = 2^u$ である．また，$\tau \in \{I, II, III, IV\}$ であり，$s_1, ..., s_k$ を，種の列とする．以降では，簡便さのため，$\mathcal{L}(\tau, k, s_1, ..., s_k)$ の $\tau, k, s_1, ..., s_k$ を省略して，単に $\mathcal{L}$ などと書くことがある．また，ハッシュ連鎖リスト $\mathcal{L}$ が与えられたとき，それに属するハッシュ連鎖の数を $k_\mathcal{L}$ と表すことがある．

2.3 ハッシュ連鎖アグリゲーション

ハッシュ連鎖リストを定義したことにより，r 個のハッシュ連鎖リストの集合として，ハッシュ連鎖アグリゲーション (HCA) $\mathcal{A}$ を定義することが出来る．

$$\mathcal{A} := \{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_r\} \quad (3)$$

$\mathcal{L}_i$ $(1 \leq i \leq r)$ は，式 (2) によって定義される．

2.4 ユーザに割り当てられるハッシュ値

次に，HCA において，ノード i に割り当てられるハッシュ値の集合を考えよう．

まず，式 (3) における任意の $\mathcal{L}$ $(\in \mathcal{A})$ について考える．$\mathcal{L}$ においては，長さ $N/k_\mathcal{L}$ $(= 2^{m-u})$ のハッシュ連鎖が $k_\mathcal{L}$ 個あるので，$\mathcal{L}$ は N 個のハッシュ値を持つ．すなわち，任意のユーザ i $(1 \leq i \leq N)$ は，$\mathcal{L}$ のあるハッシュ連鎖から，一つのハッシュ値が割り当てられる．このハッシュ値は，以下のようにして求めることができる．

まず，

$$\frac{N}{k_\mathcal{L}}(d-1) + 1 \leq i \leq \frac{N}{k_\mathcal{L}} d \quad (1 \leq d \leq k_\mathcal{L}) \ . \quad (4)$$

式 (4) を満たす d が一意に定まることに注意せよ．そこで，このような d を，$\alpha(\mathcal{L}, i)$ と書く

すると，$\mathcal{L}$ において，ユーザ i に割り当てられるハッシュ値は，

© 2017 Information Processing Society of Japan

$$(\mathcal{L}[\alpha(\mathcal{L},i)])\left[i-\frac{N}{k_{\mathcal{L}}}\cdot(\alpha(\mathcal{L},i)-1)\right]$$

となる

以上より，HCA $\mathcal{A}$ においてユーザ i に割り当てられるハッシュ値の集合は，

$$V(\mathcal{A},i):=\left\{(\mathcal{L}[\alpha(\mathcal{L},i)])\left[i-\frac{N}{k_{\mathcal{L}}}\cdot(\alpha(\mathcal{L},i)-1)\right]\;\middle|\;\mathcal{L}\in\mathcal{A}\right\}\ . \tag{5}$$

と定義される関数 V によって表現できる．KGC は，ユーザ i に対し $V(\mathcal{A},i)$ を安全なチャンネルを介して送る．ユーザ i は，送られてきた値を秘密に保持しておく．

また，$\mathcal{A}$ においてユーザ i を含むハッシュ連鎖の集合は，

$$W(\mathcal{A},i):=\{\mathcal{L}[\alpha(\mathcal{L},i)]\mid\mathcal{L}\in\mathcal{A}\} \tag{6}$$

と定義される関数 W によって表現できる．

2.5 共通鍵作成法のためのハッシュ連鎖アグリゲーション

共通鍵作成法のためのハッシュ連鎖アグリゲーション $\mathcal{HCA}$ を以下のように定義する．なお以下では，ハッシュ連鎖リスト $\mathcal{L}(\tau,k,s_1,...,s_k)$ の $\tau,k,s_1,...,s_k$ を省略する．$N=2^m$ であること注意せよ．

$$\mathcal{HCA}:=(\mathcal{L}_{(1)},\mathcal{L}_{(2)},\ldots,\mathcal{L}_{(2m-1)},\mathcal{L}_{(2m)}) \tag{7}$$

ここで，$\mathcal{L}_{(1)}=(C_N^I(s_1))$，$\mathcal{L}_{(2)}=(C_N^{II}(s_2))$，$\mathcal{L}_{(3)}=(C_N^{III}(s_3))$，$\mathcal{L}_{(4)}=(C_N^{IV}(s_4))$ であり，$i>4$ のときの $\mathcal{L}_{(i)}$ は以下のように定義される．以下では，$3\le j\le m$ である．

1. $i=2j-1$ のとき:

$$\mathcal{L}_{(2j-1)}:=(C^{III}_{2^{m-j+2}}(s_{2j-1,1}),C^{III}_{2^{m-j+2}}(s_{2j-1,2}),$$

$$\cdots,C^{III}_{2^{m-j+2}}(s_{2j-1,2^{j-2}}))$$

2. $i=2j$ のとき:

$$\mathcal{L}_{(2j)}:=(C^{IV}_{2^{m-j+2}}(s_{2j,1}),C^{IV}_{2^{m-j+2}}(s_{2j,2}),$$

$$\cdots,C^{IV}_{2^{m-j+2}}(s_{2j,2^{j-2}}))$$

ここで，整数 a,b,c について，$s_a,s_{b,c}$ はすべて異なるランダムな種を表す．

例として，$N=8$ のときの $\mathcal{HCA}=(\mathcal{L}_{(1)},\mathcal{L}_{(2)},\ldots,\mathcal{L}_{(5)},\mathcal{L}_{(6)})$ を，図 1 に示す．このとき，ユーザ 3 について考えると，$V(\mathcal{HCA},3)=\{h^3(s_1),h^6(s_2),h^7(s_3),h^2(s_4),h(s_{5,1}),h^4(s_{6,1})\}$ である．また，$W(\mathcal{HCA},3)=\{C_8^I(s_1),C_8^{II}(s_2),C_8^{III}(s_3),C_8^{IV}(s_4),C_4^{III}(s_{5,1}),C_4^{IV}(s_{6,1})\}$ である．

2.6 $\mathcal{HCA}$ による共通鍵生成

$\mathcal{HCA}$ を用いたユーザ i,j $(1\le i<j\le N)$ による，相互認証のための共通鍵作成法は以下の通りである．

1. $i+1=j$ のとき: $F(\mathcal{L}_{(1)}[1])[j],(\mathcal{L}_{(2)}[1])[i])$ とすればよい．

2. それ以外のとき: ある正整数 $(2\le)\ k_1<k_2<\cdots<k_u\ (\le m)$ を用いて，

$$\begin{aligned}&W(\mathcal{HCA},i)\cap W(\mathcal{HCA},j)=\\&\{C_N^I(s_1),C_N^{II}(s_2),\\&\mathcal{L}_{(2k_1-1)}[\alpha(\mathcal{L}_{(2k_1-1)},i)],\mathcal{L}_{(2k_1)}[\alpha(\mathcal{L}_{(2k_1)},i)],\\&\vdots\\&\mathcal{L}_{(2k_u-1)}[\alpha(\mathcal{L}_{(2k_u-1)},i)],\mathcal{L}_{(2k_u)}[\alpha(\mathcal{L}_{(2k_u)},i)]\}\end{aligned}$$

と書くことが出来る．ここで，$\alpha(\mathcal{L}_{(2k_1-1)},i)=\alpha(\mathcal{L}_{(2k_1-1)},j)=\alpha(\mathcal{L}_{(2k_1)},i)=\alpha(\mathcal{L}_{(2k_1)},j)$，$\cdots$，$\alpha(\mathcal{L}_{(2k_u-1)},i)=\alpha(\mathcal{L}_{(2k_u-1)},j)=\alpha(\mathcal{L}_{(2k_u)},i)=\alpha(\mathcal{L}_{(2k_u)},j)$ である．また，特に，$\mathcal{L}_{(2k_u-1)}[\alpha(\mathcal{L}_{(2k_u-1)},i)]$，$\mathcal{L}_{(2k_u)}[\alpha(\mathcal{L}_{(2k_u)},i)]$ は，i,j を同時に含むハッシュ連鎖の中で，その長さが最も短いものである．

このとき，ユーザ i,j は，いずれも以下の 4 個のハッシュ値を計算できる:

$$\begin{aligned}&\nu_1:=C_N^I(s_1)[j],\\&\nu_2:=C_N^{II}(s_2)[i],\\&\nu_3:=(\mathcal{L}_{(2k_u-1)}[\alpha(\mathcal{L}_{(2k_u-1)},i)])[i-\beta_{i,2k_u-1}],\\&\nu_4:=(\mathcal{L}_{(2k_u)}[\alpha(\mathcal{L}_{(2k_u)},i)])[j-\beta_{j,2k_u}]\end{aligned} \tag{8}$$

ここで，$\beta_{i,j}:=(N\cdot(\alpha(\mathcal{L}_{(j)},i)-1))/k_{\mathcal{L}_{(j)}}$ とおいた．こうして，ユーザ i,j の共通鍵 $K_{i,j}$ を，

$$K_{i,j}:=F(\nu_1,\nu_2,\nu_3,\nu_4)\ . \tag{9}$$

と計算すればよい．

2.7 セキュリティ評価

ハッシュ連鎖アグリゲーションを用いた共通鍵作成法に対して，攻撃者が一人の場合と，二人の場合について，次のようにセキュリティが評価されている．

攻撃者が一人の場合は，定理 1 を証明することができる．

定理 1. 一人の攻撃者は，任意の i,j $(1\le i<j\le N)$ について，ユーザ i,j の共通鍵 $K_{i,j}$ を計算できない．

証明については，[13] を参照されたい．

2 人以上の攻撃者が結託する場合は，定理 2 を証明することができる．

定理 2. ユーザ i,j $(1\le i<j\le N)$ について，$i<a_1\le N(\alpha-1)/k+N/2k$, $N(\alpha-1)/k+N/2k+1\le a_2<j$ を満たすユーザ a_1,a_2 は，結託してユーザ i,j の共通鍵 $K_{i,j}$ を計算することができる．ここで，α,k の定義は定理 *1* の

© 2017 Information Processing Society of Japan

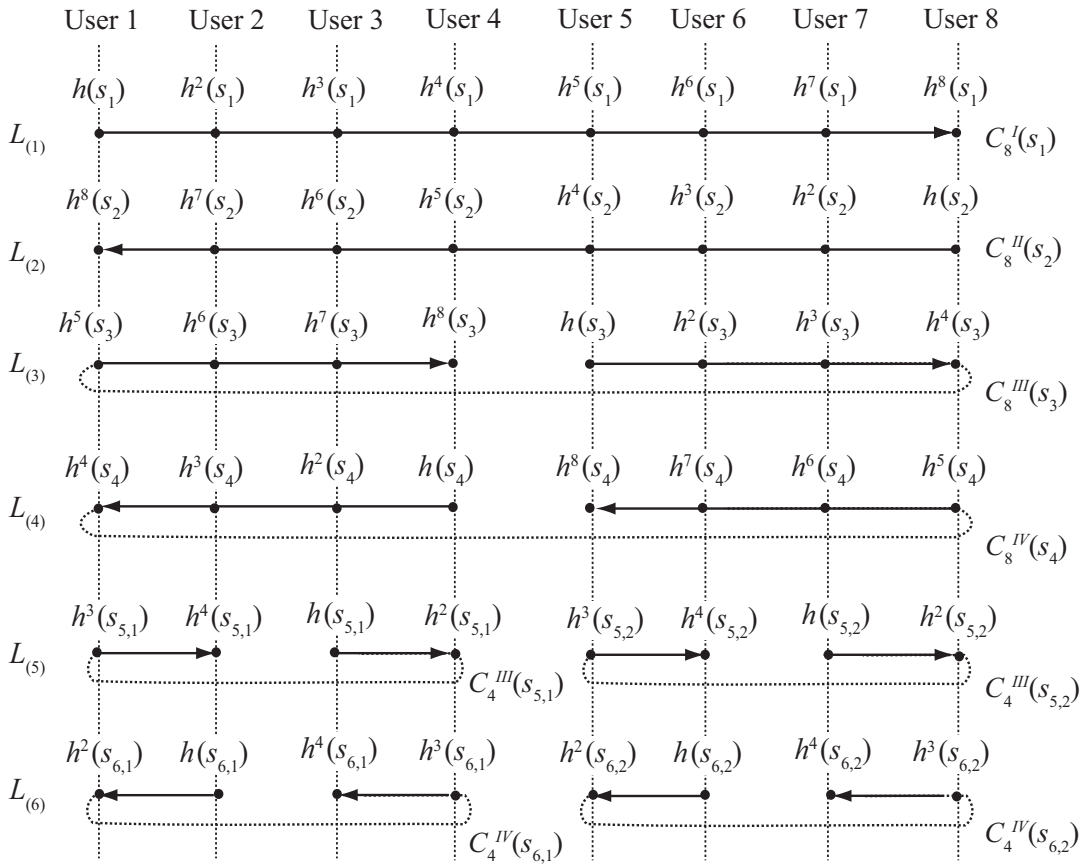

図 1 $N = 8$ のときの $\mathcal{HCA}$
Fig. 1 Our $\mathcal{HCA}$ ($N = 8$)

ものと同じである．

証明については，[13] を参照されたい．

以上より，2 人以上のユーザが結託すると，別の二人のユーザの共通鍵を計算できる状況がある．この方式は，比較的信頼される環境で使用されるべきである．

3. より詳細なセキュリティ評価

本節では，ハッシュ連鎖アグリゲーションを用いた共通鍵作成法に対して，より詳細なセキュリティ評価を行う．

3.1 それぞれのユーザから入手可能なハッシュ値の個数

ハッシュ連鎖アグリゲーションを用いた共通鍵作成法では，全てのユーザについて，それぞれ計算可能なハッシュ値の個数が同じであり，どのユーザに対しても，攻撃者になったときのリスクは同等であると考えられる．

3.2 攻撃者の人数による攻撃の危険性

本節では，攻撃者の人数によって，攻撃の難易度や危険性がどのように変化するかについて議論する．

まず，攻撃が成立する条件について整理する．ハッシュ連鎖アグリゲーションを用いて認証を行う 2 人のユーザをユーザ $i, j (i < j)$，2 人の攻撃者を，攻撃者ユーザ $a_1, a_2 (a_1 < a_2)$ とする．2.7 節に，詳細な条件が明記されているが，ここでは，例を用いて述べる．

(1) $a_1 = 3, a_2 = 5 (N = 8)$ のとき
2.7 節の条件から，攻撃が成立するのは，ユーザ i, j のどちらかが，ユーザ 4 である場合のみである．しかし，ユーザ 4 とユーザ 4 以外の全てユーザとの共通鍵を計算可能であるため，ユーザ 4 に関する認証においては，任意の相手との認証に対して，攻撃が可能である．

(2) $a_1 = 1, a_2 = 5 (N = 8)$ のとき
2.7 節の条件から，攻撃が成立するのは，ユーザ i が，2,3,4 のいずれかであり，かつユーザ j が，6,7,8 のいずれかのときである．すなわち，ユーザ 2,3,4 のいずれかのユーザと，ユーザ 6,7,8 のいずれかのユーザとの認証に対して攻撃が可能である．

以上の例から，$a_1 = 3, a_2 = 5$ のとき，攻撃可能な認証の組み合わせは 5 個であり，ユーザ 4 を含む全ての認証に対して攻撃可能，$a_1 = 1, a_2 = 5$ のとき，攻撃可能な認証の組み合わせは 9 個であり，ユーザ 2,3,4 のいずれかとユーザ 6,7,8 のいずれかの認証であれば攻撃が可能であることから，a_1 と a_2 の距離 $(a_2 - a_1)$ によって，攻撃の性質が違うことがわかる．つまり，a_1 と a_2 の距離が小さい

© 2017 Information Processing Society of Japan

ほど，a_1 と a_2 の間にいるユーザに対する攻撃の自由度が増し，a_1 と a_2 の距離が大きいほど，攻撃可能な認証のバリエーションが増加する．

次に，さらに攻撃者を追加して考えてみる．攻撃者ユーザ $a_1, a_2(a_1 < a_2)$ に，攻撃者ユーザ $a_3(a_1 < a_3 < a_2)$ を追加する．ここで，攻撃者ユーザ a_3 を含めて，攻撃者間の距離を考えると，$(a_1 < a_3 < a_2)$ であることに注意して，

$$(a_3 - a_1) < (a_2 - a_1), (a_2 - a_3) < (a_2 - a_1)$$

となるため，攻撃者の数が増加する毎に，それぞれのユーザに対する攻撃の自由度が増加し，より攻撃として強力になると考えられる．

4. ハッシュ連鎖アグリゲーションの応用

$\mathcal{HCA}$ において，$N = 2^m$ (m :正整数)　を満たすとき，各ユーザに，ハッシュ連鎖の 1 番目のハッシュ値が必ず割り当てられる．ハッシュ連鎖の 1 番目のハッシュ値は，その値を割り当てられたユーザ以外からは計算することが出来ないため，このハッシュ値を用いて各ユーザの秘密を作成することが出来る．以下では，これを用いた $\mathcal{HCA}$ の応用について議論する．

4.1 キーエスクロー

キーエスクローは，鍵供託とも呼ばれ，鍵を，信頼できる機関に預け，特定の状況下において，許可された第 3 者がその鍵にアクセス出来る仕組みのことである．Joye らは，one-way cross trees (OWCT) と呼ばれる手法を提案し，その応用として，OWCT を用いたキーエスクローを提案した．しかし，Joye らの，OWCT を用いたキーエスクローでは，複数回の鍵の開示要求に対応出来ない場合が存在する．我々は，$\mathcal{HCA}$ を用いたキーエスクローを提案する．$\mathcal{HCA}$ を用いたキーエスクローでは，あるユーザが，ある通信に使用するための秘密鍵を毎日更新するような場合を想定し，特定の期間の通信の解析のために，警察等の組織から，その期間の秘密鍵を開示する要求があった場合を考える．$\mathcal{HCA}$ を用いたキーエスクローでは，複数回の鍵の開示要求に対応することが可能である．以下，$\mathcal{HCA}$ を用いたキーエスクローについて説明する．

まず，次のように，毎日異なる秘密鍵を作成することが出来る．

i 日目の秘密鍵 : $F(V(\mathcal{A}, i))$ (F :一方向性関数)

ここで，i 日目から j 日目までの秘密鍵の開示を要求されたとする．このとき，$V(\mathcal{A}, i)$ 及び，$V(\mathcal{A}, j)$ を公開し，さらに，i 日目から j 日目までのハッシュ連鎖の 1 番目のハッシュ値を公開することで，i 日目から j 日目までの秘密鍵を，第 3 者が計算することが出来る．また，これらの値を公開しても，i 日目から j 日目以外のハッシュ連鎖の 1 番目のハッシュ値を計算出来ないため，i 日目から j 日目までの期間以外の秘密鍵が計算されることはない．すなわち，複数回の秘密鍵の開示要求に対応することが出来る．

次に，例を用いて説明する．尚，1 を参照すると良いだろう．鍵を 8 日更新する場合，$N = 8$ の $\mathcal{HCA}$ を使用する．1 日目の秘密鍵は，

$$F(h(s_1), h^8(s_2), h^5(s_3), h^4(s_4), h^3(s_{5,1}), h^2(s_{6,1}))$$

となる．そして，3 日目から 5 日目までの期間の秘密鍵の開示要求があったとすると，$V(\mathcal{A}, 3)$ 及び，$V(\mathcal{A}, 5)$，すなわち，

$$h^3(s_1), h^6(s_2), h^7(s_3), h^2(s_4), h(s_{5,1}), h^4(s_{6,1})$$

と

$$h^5(s_1), h^4(s_2), h(s_3), h^8(s_4), h^3(s_{5,1}), h^2(s_{6,1})$$

を公開し，さらに，4 日目に使用したハッシュ値の中から，ハッシュ連鎖の 1 番目の値である $h(s_4)$ を公開する．4 日目に使用したハッシュ値のうち，$h(s_4)$ 以外のハッシュ値は，$V(\mathcal{A}, 3)$ 及び，$V(\mathcal{A}, 5)$ から計算可能であるため，上述の 17 個のハッシュ値を公開すればよい．また，開示要求のあった期間以外については，公開したハッシュ値から，2 日目の，$h(s_{6,1})$ や，6 日目の，$h(s_{6,2})$ などを計算出来ないため，他の日の秘密鍵が計算されることはない．

5. 結論

本論文では，全く新しく，かつ柔軟なハッシュ連鎖構成法 “ハッシュ連鎖アグリゲーション” とそれを用いた，二者間での共通鍵生成法に対して，より詳細なセキュリティ評価を行い，さらなる応用について議論した．

より詳細なセキュリティ評価によって，どのユーザが攻撃者であっても，攻撃の難易度が変化しないこと，攻撃者の数の増加に伴い，それぞれのユーザに対する攻撃の自由度が増加し，より攻撃として強力になることを明らかにした．また，“ハッシュ連鎖アグリゲーション”　のさらなる応用として，キーエクスローを提案し，より一層，手法の有用性が高まったと考えている．今後の課題は，“ハッシュ連鎖アグリゲーション” を用いた，新しい認証方式を提案することである．

謝辞

本研究は科学研究費補助金 (課題番号 JP15K00189) の助成，および国立研究開発法人科学技術振興機構（JST）の国際科学技術協力基盤整備事業の支援を受けたものである．

参考文献

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

© 2017 Information Processing Society of Japan

[2] J. Katz and Y. Lindell, *Introduction to Modern Cryptography: Principles and Protocols.* Chapman and Hall/CRC, 2007.

[3] D. J. Bernstein, J. Buchmann, and E. Dahmen, Eds., *Post-Quantum Cryptography.* Springer-Verlag, 2009.

[4] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[5] R. Dutta, E. Chang, and S. Mukhopadhyay, "Efficient self-healing key distribution with revocation for wireless sensor networks using one way key chains," in *Proceedings of 5th International Conference on Applied Cryptography and Network Security (ACNS)*, ser. Lecture Notes in Computer Science, no. 4521. Springer-Verlag, 2007, pp. 385–400.

[6] M. Joye and S. Yen, "One-way cross-trees and their applications," in *Public Key Cryptography (PKC)*, ser. Lecture Notes in Computer Science, vol. 2274. Springer-Verlag, 2002, pp. 346–356.

[7] L. Lamport, "Password authentication with insecure communication," vol. 24, no. 11, pp. 770–772, Nov. 1981.

[8] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in *Proceedings of the IEEE Symposium on Security and Privacy*, May 2000, pp. 56–73.

[9] R. Rivest and A. Shamir, "PayWord and MicroMint: Two simple micropayment schemes," in *Security Protocols*, ser. Lecture Notes in Computer Science, vol. 1189. Springer-Verlag, 1997, pp. 69–87.

[10] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Oct. 2008.

[11] X. Guo, S. Huang, L. Nazhandali, and P. Schaumont, "Fair and comprehensive performance evaluation of 14 second round SHA-3 ASIC implementations," NIST 2nd SHA-3 Candidate Conference, Aug. 2010.

[12] D. Coppersmith and M. Jakobsson, "Almost optimal hash sequence traversal," in *Proceesings of 6th International Conference on Financial Cryptography (FC 2002)*, ser. Lecture Notes in Computer Science, vol. 2357. Springer-Verlag, 2002, pp. 102–119.

[13] Y. Kurihara and M. Soshi, "A Novel Hash Chain Construction for Simple and Efficient Authentication," Privacy, Security and Trust, Dec.2016.

© 2017 Information Processing Society of Japan