効率的な耐故障性をもつ鍵共有方式

波多野 哲也 †　　宮地 充子 †

† 北陸先端科学技術大学院大学 情報科学研究科
923-1292 石川県能美市旭台 1-1

{s0810048, miyaji}@jaist.ac.jp

あらまし　インターネット上で複数のプレーヤーが秘匿通信をしたい場合に，各プレーヤーに対して効率的に鍵を配布する必要がある．任意のプレーヤーの故障などでプロトコル実行に失敗すると鍵共有できない．そこで幾つかのプレーヤーがプロトコル実行に失敗することに対して耐性を持つ鍵共有プロトコルが提案されている．本研究では効率的な通信量や計算量を保ち耐故障性を持った鍵共有プロトコルを提案する．

Efficient Group Key Agreement Robust against Some Node Faults

Tetsuya Hatano†　　Atsuko Miyaji†

Japan Advanced Institute Science and Technology(JAIST)
1-1, Asahidai, Nomi-shi, Ishikawa 923-1292, Japan

{s0810048, miyaji}@jaist.ac.jp

Abstract　Efficient key distribution is necessary for anonymous communication among multiple nodes. In case of any node is not available, protocol has to be run again which is the cause of low performance. Robust key distribution schemes which can deal with the above cases have been proposed[5]. In the sensor network environment, it is important and necessary for group key agreement protocol to run in low computation cost and message exchange cost. Here we propose a robust group key agreement protocol scheme which is able to run in relatively low computation and message exchange cost compared to the previous research works.

1　関連研究

耐故障性のない n プレーヤーのグループ鍵共有プロトコルではあるプレーヤーが故障などでプロトコル実行に失敗すると鍵共有できない．Amier らは最初に耐故障性を持ったグループ鍵共有プロトコルを提案した [1]．彼らは Steiner らによって提案された鍵共有プロトコルに Group Communication System(GCS) を組み合わせることによって実現した．GCS はグループ鍵共有プロトコル間のプレーヤーの端末故障を見つけることができるのでそれに応じて対処することができる．しかしラウンド数が $O(n)$ で通信量は $O(n^2)$ 必要である．同様に Cachin と Strobl(CS) は非同期なネットワーク上で動作するラウンドが一定な耐故障性を持った鍵共有プロトコルを提案した [3]．しかし信頼できるブロードキャストチャネルを仮定するために，CS プロトコルのラウンド数は 2 ラウンドで各々のプレーヤーは $O(n)$ のメッセージをブロードキャストし，$O(n)$ の鍵計算を行う．Jarecki らが Burmester らに

よって提案されたグループ鍵共有プロトコル [2] に基づき 2 ラウンドで耐故障性を持った鍵共有プロトコルを提案した [5]．t 個のプレーヤーの故障に対して $O(t)$ のメッセージをブロードキャストすることで t-耐性を持つ．本研究の内容は以下の通りである．2 章では準備として GKA の安全性と性能について説明する．3 章では既存研究であるリング構造に基づく耐故障性を持った鍵共有法と木構造に基づく鍵共有法について説明する．4 章では提案方式について説明する．5 章では評価について説明する．次に 6 章では安全性について説明する．最後に 7 章でまとめを行う．

2 準備

2.1 GKA の安全性

Katz-Yung は能動攻撃に対して安全なグループ鍵共有に変換するコンパイラーを提案した [6]．また Yvo らが Katz-Yung のコンパイラー [6] を使い，BDII グループ鍵共有が能動攻撃に対して安全であることを示した．[4]．本研究では受動攻撃のケースに焦点を当てて考えていく．

定義 1. U_1，U_2，$\cdots$，U_n をプロトコル P に参加している多項式時間で計算するチューニングマシンとする．プロトコル P とはグループ鍵共有 *(GKA)* プロトコルであり，全てのグループプレーヤーがあるプロトコルを実行したとき各々のプレーヤー U_i が同じ鍵 $K = K_i$ を計算できることである．このとき U_i をプレーヤーと呼び，グループを n プレーヤーとする．

定義 2. プロトコル P を n プレーヤーに対するグループ鍵共有 *(GKA)* とするとき，k をセキュリティパラメータとする．また A を外部の受動攻撃者とし，プロトコル P のインスタンスを持っていると仮定する．このときインスタンスによって作られた鍵 K と同じ長さのランダムなビット列を $\frac{1}{2}\epsilon$ より小さい確率 *(ϵ はパラメータ kで無視できる確率)* で区別できない場合，プロトコル P は受動攻撃に対して安全という．

定義 3. $\mathbb{G}$ を位数 p の群とするとき，$\mathbb{G}\in g, y$ に対して $\{g^a, g^b, g^{ab}\}$ と $\{g^a, g^b, y\}$ を区別する問題を *DDH* 問題という．

2.2 性能

グループ鍵共有の効率性について述べる．プロトコルの効率性に関係するものを以下に示す．
ラウンド数: ラウンド数とは鍵共有プロトコルが終了するまでのラウンド数である．
通信量: 任意の 1 プレーヤーによって送付された最大メッセージ数である．メッセージは決められた受信者に送付する point to point と全員にメッセージを送付する Broadcast に分けて考える．Broadcast チャネルを仮定し，プレーヤーは Broadcast チャネルにメッセージを送る．なおマルチキャストとブロードキャストは区別しない．
計算量: 1 プレーヤーの計算量である．このとき EX をベキ演算，M を乗算とする．
故障数: 最大故障可能な個数

3 既存研究

3.1 リング構造に基づく耐故障性を持った鍵共有 [5]

この章では既存研究であるリング構造に基づく耐故障性を持った鍵共有 [5] を紹介する．信頼できるブロードキャストチャネルを仮定すると任意の数の障害に対して耐性を持つことができる．しかし各々のプレーヤーがブロードキャストするメッセージ数はプレーヤーの数に比例する．対照的に通常のグループ鍵共有プロトコルではブロードキャストするメッセージ数を一定にすることができる．この提案された方式は任意の $t < n$ に対して t 個のプレーヤーの端末故障に対して耐性のある鍵共有プロトコルである．この耐故障性をもったグループ鍵共有プロトコルは Burmester らによって提案された BD グループ鍵共有 [2] に基づいている．初めに耐故障性をもたせるために BD 方式を修正するので BD 方式を説明する．このとき P_i をプレーヤー $i \in [1, n]$ とする．

プロトコル 1. 2 ラウンド *BD* グループ鍵共有 *[2]*

Round 1*:* プレーヤー P_i は秘密に選んだ $r_i \in \mathbb{Z}_p$ に対して，$z_i = g^{r_i}$ を計算し，ブロードキャストする

Round 2*:* プレーヤー P_i は z_{i-1}，z_{i+1} を取得し，$x_{[\mathtt{i-1,i,i+1}]}$ をブロードキャストする

$$\begin{aligned} x_{[\mathtt{i-1,i,i+1}]} &= \left(\frac{z_{i+1}}{z_{i-1}}\right)^{r_i} \\ &= g^{r_i r_{i+1} - r_{i-1} r_i} \end{aligned}$$

鍵計算*:* プレーヤー P_i は鍵 K_i を以下のように計算し，共有鍵 K にする．

$$\begin{aligned} K_i &= (z_{i-1})^{nr_i} \cdot x_{[\mathtt{i-1,i,i+1}]}^{n-1} \cdot x_{[\mathtt{i,i+1,i+2}]}^{n-2} \cdots \\ &\quad x_{[\mathtt{i-3,i-2,i-1}]} \\ &= g^{r_1 r_2 + r_2 r_3 + \cdots + r_n r_1} = K \end{aligned}$$

メッセージ $x_{[\mathtt{i-1,i,i+1}]}$ をガジェット値と呼ぶ．二つのガジェット値 $x_{[\mathtt{i-1,i,i+1}]}$ と $x_{[\mathtt{i,i+1,i+2}]}$ は $x_{[\mathtt{i-1,i,i+1}]} \cdot x_{[\mathtt{i,i+1,i+2}]} = g^{r_{i+1} r_{i+2} - r_{i-1} r_i}$ となり，項数が 2 個のままであるとき連結可能という．連結可能なグラフを構成すると共通のセッション鍵 K を作成することができる．BD 鍵共有が耐故障性を持たない理由は，1 プレーヤーの端末の故障でもガジェット値が欠如し，リング構造が壊れるからである．故障したプレーヤーの端末があったとしても，正常なプレーヤーの端末でガジェット値が連結可能になるように木構造を構成すれば共通のセッション鍵 K を作成することができる．そこで二重ノードと鍵の再利用でガジェット値の数を n に抑え，プレーヤーの耐故障性を実現した RGKA 方式を記述する．

プロトコル 2. *n* 耐故障性を持った鍵共有 *(RGKA)*

Round 1*:* プレーヤー P_i は秘密に選んだ $r_i \in \mathbb{Z}_p$ に対して，$z_i = g^{r_i}$ を計算し，ブロードキャストする

Round 2*:*

- *ActList(*全てのプレーヤーのインデックスリスト*)* を用意する
- プレーヤー P_i は $k \in$ *ActList* に対して以下を計算する．ただし $x_{[\mathtt{k,i,i'}]} = (x_{[\mathtt{i,i',k}]})^{-1}$ である．

$$x_{[\mathtt{k,i,i'}]} = \left(\frac{z_i}{z_k}\right)^{r_i}$$

- $\{x_{[\mathtt{k,i,i'}]}\}_{k \in ActList}$ をブロードキャストする．

鍵計算*:* *ActList* を *Round* 2 を完了したプレーヤーのインデックスリストに更新する．そして *ActList* に載っているプレーヤー P_i を並べる $\{P_{a_1}, P_{a_2}, \cdots P_{a_m}\}$．$P_{a_i}$ は鍵 K_{a_i} を以下のように計算し，共有鍵 K にする．ここで $X_{a_i} = x_{[\mathtt{a_{i-1},a_i,a_{i'}}]} \cdot x_{[\mathtt{a_i,a_{i'},a_{i+1}}]} = g^{r_{a_i} r_{a_{i+1}} - r_{a_{i-1}} r_{a_i}}$ とする．

$$\begin{aligned} K_{a_i} &= z_{a_{i-1}}^{m \cdot r_{a_i}} \cdot X_{a_i}^{m-1} \cdot X_{a_{i+1}}^{m-2} \cdot \cdots \cdot X_{a_{i-2}} \\ &= g^{r_{a_1} r_{a_2} + r_{a_2} r_{a_3} + \cdots + r_{a_m} r_{a_1}} = K \end{aligned}$$

上記のプロトコルではプレーヤー P_i は最大 $n-1$ 個のガジェット値をブロードキャストしていた．これを $t < n$ 個だけをブロードキャストして t 個のノードの遮断に耐性をもったリング構造に基づくグループ鍵共有 (t-RGKA) を構築することができる．

3.2 木構造に基づく鍵共有 [2]

初めにプレーヤーの配置について説明する．ϵ を長さ 0 の空の文字列とし，$\{0,1\}^l$ は長さ l の文字列の集合とする．このとき $\{0,1\}^{<l} = \bigcup_{0 \le i < l} \{0,1\}^i$, $\{0,1\}^{\le l} = \bigcup_{0 \le i \le l} \{0,1\}^i$ と定義する．またある文字列 $\omega \in \{0,1\}^l$ があるとき，文字列 ω の右側に 0 を追加した文字列を $\omega 0$，1 を追加した文字列を $\omega 1$ とする．このとき木の各プレーヤーに $\omega \in \{0,1\}^l$ の ID 番号を割り振り，各プレーヤーを P_ω とする．割り振りかたはあるプレーヤー P_ω があるとき，左の子を $P_{\omega 0}$，右の子を $P_{\omega 1}$ とし，再帰的に ID 番号を割り振る．

以上のことを行い，プレーヤーを配置する．また ω の先頭 i ビットを $\omega|_i$ と表記する．このとき $\omega|_0 = \epsilon$ とする．ここで既存研究である木構造に基づく BDII[2] に注目する．BDII は以下のように与えられる．このとき我々はある高さ

l のプレーヤー P_w の親を $\mathsf{par}(\omega) = \omega_{|l-1}$，左の子を $\mathsf{l.child}(\omega) = \omega 0$，右の子を $\mathsf{r.child}(\omega) = \omega 1$，先祖を $\mathsf{ancestor}(\omega) = \bigcup_{0 \le i \le l-1} \omega|i$ とする．

プロトコル 3. 2 ラウンド *BDII* 鍵共有 *[2]*
Round 1*:* ある高さ l のプレーヤー P_ω は秘密に選んだ $r_\omega \in \mathbb{Z}_p$ に対して，$z_\omega = g^{r_w}$ を計算し，親と子に送信する
Round 2*:* プレーヤー P_ω は $x_{left,\omega}$ と $x_{right,\omega}$ を計算し，左と右の子孫にマルチキャストする

$$x_{left,\omega} = (\frac{z_{\mathtt{par}(\omega)}}{z_{\mathtt{l.child}(\omega)}})^{r_\omega} = (\frac{z_{\omega|l-1}}{z_{\omega 0}})^{r_\omega}$$

$$x_{right,\omega} = (\frac{z_{\mathtt{par}(\omega)}}{z_{\mathtt{r.child}(\omega)}})^{r_\omega} = (\frac{z_{\omega|l-1}}{z_{\omega 1}})^{r_\omega}$$

鍵計算*:* プレーヤー P_ω は鍵 K_ω を以下のように計算し，共有鍵 K とする．

$$\begin{aligned} K_\omega &= (z_{\mathtt{par}(\omega)})^{r_\omega} \cdot \Pi_{j \in ancestor(\omega)} x_j \\ &= (z_{\omega|l-1})^{r_\omega} \cdot \Pi_{0 \le i \le l-2} x_{\omega|i.\omega|i+1.\omega|i+2} \\ &= K \end{aligned}$$

4 提案方式

3.2 章で記載した BDII 鍵共有 [2] は一つのプレーヤーの故障で，そのプレーヤーの全ての子孫ノードが鍵共有できないという問題が生じる．このような問題に対し，耐故障性を持ちかつ t-RGKA より通信量や計算量を減らした鍵共有法を提案する．

4.1 提案の概要

まず簡単に提案方式の概要を述べる．2 つの木を独立的な関係から相互に親戚関係へ設定する．BDII では root プレーヤーの 0 と 1 は互いに親の関係であるが，それぞれの子孫とは関係を定義しない．これに対して，本方式では，0 と 1 は互いに親の関係だがそれぞれの子にとっては，おばの関係になる．図 1 では root0 と root1 の関係を表す．以上のことを行い，木構造を構成すると図 1 になる．

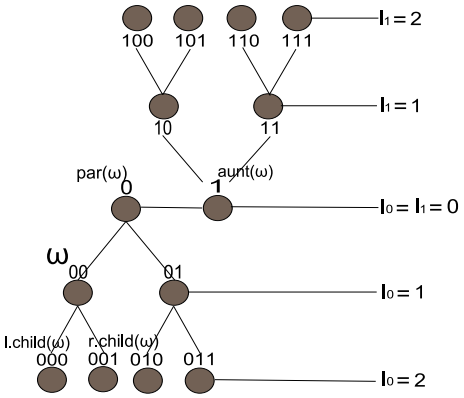

図 1: $n = 14$ のときの root0 と root1 の関係

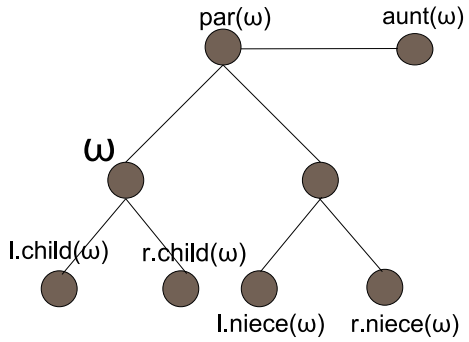

図 2: プレーヤー P_ω の関係図

4.2 兄弟補助による $\frac{n}{2}$-耐故障性を実現した鍵共有

兄弟プレーヤーが補助し合うことにより，あるプレーヤーが故障しても兄弟プレーヤーが故障していなければその子孫の鍵共有を実現する．高さ l_0 のプレーヤー P_ω に対して，親を $\mathsf{par}(\omega)$，左の子を $\mathsf{l.child}(\omega)$，右の子を $\mathsf{r.child}(\omega)$，叔母を $\mathsf{aunt}(\omega)$，左の甥を $\mathsf{l.niece}(\omega)$，右の甥を $\mathsf{r.niece}(\omega)$ となる．以上の関係図を説明すると図 2 となる．

プロトコル 4. $\frac{n}{2}$-耐故障性をもつ鍵共有法
Round 1*:* プレーヤー P_ω は秘密に選んだ $r_\omega \in \mathbb{Z}_p$ に対して，$z_\omega = g^{r_\omega}$ を計算し，親と叔母，子，甥に送信する．
Round 2*:*

- *ActList* を用意する．
- プレーヤー P_ω は以下を計算し，子孫にマ

ルチキャストする

$$x_{[\mathtt{par}(\omega),\omega,\omega']} = \left(\frac{z_{\mathtt{par}(\omega)}}{z_\omega}\right)^{r_\omega}$$

$$x_{[\mathtt{aunt}(\omega),\omega,\omega']} = \left(\frac{z_{\mathtt{aunt}(\omega)}}{z_\omega}\right)^{r_\omega}$$

$$x_{[\omega,\omega',\mathtt{l.child}(\omega)]} = \left(\frac{z_\omega}{z_{\mathtt{l.child}(\omega)}}\right)^{r_\omega}$$

$$x_{[\omega,\omega',\mathtt{r.child}(\omega)]} = \left(\frac{z_\omega}{z_{\mathtt{r.child}(\omega)}}\right)^{r_\omega}$$

$$x_{[\omega,\omega',\mathtt{l.niece}(\omega)]} = \left(\frac{z_\omega}{z_{\mathtt{l.niece}(\omega)}}\right)^{r_\omega}$$

$$x_{[\omega,\omega',\mathtt{r.niece}(\omega)]} = \left(\frac{z_\omega}{z_{\mathtt{r.niece}(\omega)}}\right)^{r_\omega}$$

鍵計算: *ActList* を *Round* 2 を完了したプレーヤーのインデックスリストに更新する．プレーヤー P_ω は *ActList* を使い，木構造を再構築する．再構築されたある高さ l_j の先祖プレーヤー $P_j \in \overline{ancestor}(\omega)$ に対して，$\mathit{ActList}_{l_{j-1}}$ *(*親 *par(*j*)*，叔母 *aunt(*j*))* と $\mathit{ActList}_{l_{j+1}}$ *(*左の子 *l.child(*j*)*，右の子 *r.child(*j*)*，左の甥 *l.niece(*j*)*，右の甥 *r.niece(*j*))* としたとき，$k \in \mathit{ActList}_{l_{j-1}}$，$m \in \mathit{ActList}_{l_{j+1}}$ に対して，$X_j = x_{[k,j,j']} \cdot x_{[j,j',m]}$ とする．このとき P_ω は鍵 K_ω を以下のように計算し，共有鍵 K にする．

$$K_\omega = (z_{\mathtt{par}(\omega)})^{r_\omega} \cdot \Pi_{j \in \overline{ancestor}(\omega)} X_j = K$$

5 評価

性能の評価を行う．表 1 の提案方式と RGKA ($t = 2$) の場合で比較できる．通信量は Broadcast の場合と変わらないが，決められた受信者に送付する point to point の場合は $O(\log_2 n)$ になっていることがわかる．また計算量である乗算回数も $O(\log_2 n)$ になる．故障数は，共に約半分の故障に耐性を持っている．

6 安全性

定理 1. *DDH* 問題が難しいと仮定すると，提案方式は安全なグループ鍵プロトコルであることを示す．すなわち下記を示せばよい．

$$Adv_P^{KE} \leq Adv_{\mathbb{G}}^{ddh}(t')$$

ここで Adv_P^{KE} は提案方式に対する攻撃者であり，t 時間で *Execute* クエリに質問することができる．また $t' = t + O(|P|q_{ex}t_{exp})$ で，t_{exp} はベキ演算を行うのに必要な時間である．ここで留意すべきことは t_{exp} の時間は提案方式の実行時間以内である．

Proof. ここで提案方式に対しての攻撃者 A を仮定し，攻撃者 A は t 時間で Execute クエリに質問することができる．このとき DDH 問題に対して識別できる D が存在することを示す．最初に攻撃者 A が Execute クエリーをつくるケースを考える．D に組 $(g_1, g_2, g_3) \in \langle g \rangle^3$ を与える．このとき D が正当な複写を A に対して次のように作ることができる．そして D は A を実行し，$z = xy$ かそうではないか決める．もし 1 が出力された場合 (g_1, g_2, g_3) は正当な Diffie-Hellman 組である．$z'_1 = g_1$，$z'_2 = g_2$ とおく．そしてランダムに c_3，c_4，$\cdots$，$c_n \in_R \mathbb{Z}_p$ を選び，$i \geq 3$ に対して，$z'_i := z'^{c_i}_{parent(parent(i))}$ とおく．例を挙げると $z'_3 = g_2 \cdot g^{-c_3}, z'_5 = g_1 \cdot g^{-c_5}$ などとなる．$i \geq 3$ に対して x'_i は $x'_i := (z'_{parent(i)})^{c_i}$ で求めることができる．$c_i (i \geq 3)$ がランダムな一様分布であるとき，z'_i と x'_i は識別できない．複写を $T = (z'1, z'_2, \cdots, x'_1, x'_3, \cdots, x'_n)$ とする．Teat クエリーを要求した上で，D は $sk' = g_3$ を出力する．実際にもし sk' が正当な共有鍵なら $g_3 = sk' = z_1'^{\log_g z'_2} = g_1^{log_g g_2}$ となる．すなわち (g_1, g_2, g_3) は正当な Diffie-Hellman 組である．従って D は A と同様なアドバンテージで成功する．そして複写を生成するためにベキ演算の $(2n-4)t_{exp}$ を加える必要があり，そしてグループのノードの数倍する． □

7 まとめ

プレーヤーがプロトコルの実行に失敗することに対して，耐性を持った鍵共有プロトコルを

表 1: 提案方式と既存方式のラウンド数と通信量，計算量の比較

	ラウンド	通信量		計算量		故障数
		Broadcast	point to point	EX	M	
BD	2	2	$2n$	3	n	1
RGKA	2	$n+1$	$n(n+1)$	$n+1$	$2n$	n
t-RGKA	2	$2t+1$	$(2t+1)n$	$2t+2$	$2n$	$n(t-1)/t$
提案方式	2	7	$6\log_2 n+6$	8	$2\log_2 n$	$n/2$

紹介した．そして既存研究より計算量や通信量を減らしたプロトコルを提案し，評価を行った．

参考文献

[1] Y. Amir, Y. Kim, C. Nita-Rotaru, J. Schultz, J. Stanton, and Gene Tsudik, “Exploring Robustness in Group Key Agreement”, In Proc. 21st IEEE International Conference on Distributed Computing System, pp 399-409, 2001.

[2] M. Burmester and Y. Desmedt, “A secure and efficient conference key distribution system”, In Advances in Cryptology - EUROCRYPT ’94, 1995.

[3] C. Cachin and R. Strobl, “Asynchronous Group Key Exchange with Failures”, In Proc. 23rd ACM Symposium on Principles of Distributed Computing (PODC 2004), pp 357-366 July 2004.

[4] Y. Desmedt, Tamja Lange and Mike Burmester, “Scalable Authenticated Tree Based Group Key Exchange for Ad-Hoc Group”, ITSC Bochum 2004.

[5] Stanislaw Jarecki, Jihye Kim and Gene Tsudik, “Robust Group Key Agreement Using Short Broadcast”, In Proc. of ACM CCS 2007, pp 411-420 ACM, 2007.

[6] J. Katz and M. Yung, “Scalable Protocols for Authenticated Group Key Exchange”, In CRYPTO 2003, LNCS2729(2003), 110-125, Springer-Verlag.