Vol.2010-CSEC-50 No.46
2010/7/2

高効率 3 パーティ秘匿関数計算の情報理論的安全性

五十嵐 大[†1] 千田浩司[†1] 高橋克巳[†1]

本稿では筆者らが第 48 回 CSEC 研究会にて提案した, 効率的な 3 パーティ秘匿関数計算プロトコルの情報理論的な安全性について議論する. 本プロトコルは加減算, 乗算及び 2 進変換をサポートすることで, 論理回路の汎用性を持ちながらも算術演算を効率よく処理することができる手法である. 本稿では本プロトコルが上記汎用性, 効率性に加え semi-honest モデルにおける完全秘匿性を持つことを証明する.

Information-theoretic Security Analysis of the Efficient 3-Party Secure Function Evaluation

Dai Ikarashi,[†1] Koji Chida[†1]
and Katsumi Takahashi[†1]

In this report, we discuss information-theoretic security of an efficient 3-party secure function evaluation protocol which we have proposed in 48th workshop of CSEC Group. The protocol has both general versatility and efficiency on arithmetic operations due to its supporting adding, subtracting, multiplying, and binary decomposition. Moreover, we show that it has *perfect secrecy* in the semi-honest model, in addition to these two properties.

1. はじめに

近年，個人に関する様々な情報を容易に取得できる環境の進歩が目覚ましいが，個人に関する情報の利活用については，個人情報保護・プライバシの観点から，国内外で法制度・ガイドラインや標準化の整備と併せた技術的対策が検討されている[1)–5)]．そして技術的対策に注目すると，いわゆる広義の匿名化が有効な手段として検討が進められている．ここで広義の匿名化とは，個人に関する情報が個人と結び付くことの無いように情報を加工する手段全般を指す．しかし一般に匿名化による対策は，情報の利用が限定的になることに加え，個人に関する情報が個人と結び付かないことの保証が容易では無い．その理由の一つとして，個人に関する情報と個人の結び付けが，予め備えている知識や情報に大きく左右されることが挙げられる．例えば，氏名，住所，年齢，性別，職業，購買履歴（日時，場所，商品名）からなるデータをマーケティング用途に氏名，住所を ID 番号に置き換えて第三者提供や一般公開を行った場合，年齢，性別，職業，および一部の購買履歴から ID 番号に対応する個人を特定できる者がいないとも限らず，その ID 番号から特定の個人の購買履歴が全て紐付いてしまう可能性があり，これはプライバシの観点から望ましくない．このように，個人に関する情報をどの程度まで開示可能かという問題は，情報の利活用と保護の両方の観点から慎重に考える必要がある．

上述の開示問題の有力な解決手段として，各種計算の入力となるデータを秘匿しつつ計算結果を求める**秘匿関数計算**（Secure Function Evaluation）[6)] が近年注目されている．特に情報を保護しつつデータマイニングを行う**プライバシ保護データマイニング**（Privacy-Preserving Data Mining）技術は，Lindell, Pinkas による研究成果[7)] を端緒に秘匿関数計算を利用した手法が数多く提案されている[8),9)]．しかしながら秘匿関数計算は一般に通常の関数計算と比べ処理時間が著しく増加し，特にデータマイニングは入力データ数や計算量が膨大となる場合があるため，秘匿関数計算を利用する場合は処理時間の軽減が特に大きな課題となる．なお本稿では，情報を保護しつつ統計処理を行うための技術も合わせてプライバシ保護データマイニングと呼ぶことにする[⋆1]

本稿でプライバシ保護データマイニング技術として利用可能とすることを目指して筆者らが第 48 回 CSEC 研究会にて提案した[10)], 軽量かつ汎用型の秘匿関数計算方式 (以降 **CIC48 方式**と呼ぶ) の安全性について議論する．当方式は 3 主体の協調計算により入力値を秘匿しつつ各種計算を効率良く実行し，2 主体の取得情報を得ない限り入力値を秘匿できる特徴を持つ方式として提案したが, その秘匿性の証明は 10) では与えなかった. そこで筆者らは本稿においてこの証明を与え, 本方式の安全性を数学的に確かなものとする. 具体的には, 本方式がパーティ間の結託を許さない semi-honest モデルにおいて完全秘匿性を持つことを

†1 NTT 情報流通プラットフォーム研究所
NTT Information Sharing Platform Laboratories

⋆1 最近ではより広範の技術を指す，プライバシ保護データ分析（Privacy-Preserving Data Analysis）やプライバシ保護データ活用（Privacy-Preserving Data Utilization）といった用語も見受けられる．

示す.

以降，2 節で関連研究を紹介し，3 節で準備として本稿で安全性を議論する対象である CIC48 方式と, 情報理論的安全性及び完全秘匿について紹介する．そして 4 節で本稿の主題である CIC48 方式の完全秘匿性証明を行い, 最後に 5 節で本稿をまとめる．

2. 関 連 研 究

秘匿関数計算は一般に，計算対象の入力値を秘匿処理したうえで提供する主体と，その入力値を復元すること無く処理する主体の少なくとも 2 主体が関与し，秘匿方法は暗号化や秘密分散が代表的である．具体的な実現方法は Yao によって提案された[11]．これは論理回路演算を実行可能な状態のまま秘匿する主体と，その秘匿された論理回路演算を実行する主体によって構成され，実行には複数主体の協調計算が必要なことから**マルチパーティプロトコル**とも呼ばれる (2 主体に特化した方式は区別して **2 パーティプロトコル**と呼ぶ場合が多い)．論理回路演算を実行するマルチパーティプロトコルとして，紛失通信（Oblivious Transfer）を利用した方式[12] や，MIX-net を利用した方式[13] 等が提案されている．なお論理回路演算の実行を可能とする秘匿関数計算は**秘匿回路計算** (Secure Circuit Evaluation) と呼ばれる．特に最近では秘匿回路計算の実行を単独で行うことを可能とする準同型暗号[14] が注目を集めている．また Yao の方式に基づく，2 パーティプロトコル[15] や計算委託型の 2 パーティプロトコル[16] 等の実装報告も見られるようになり，実用に向けた動きが加速しつつある．ここで計算委託型とは，入力提供主体と計算主体が異なる主体構成を指し，2 パーティであれば計算主体が 2 主体であることを意味する．

一方，環 $\mathbb{Z}/m\mathbb{Z}$（m は適当な整数）上の算術演算を可能とする秘匿関数計算も提案されており，秘密分散に基づく方式[17),18)] や準同型暗号に基づく方式[19),20)] が知られている．なお秘匿回路計算は一般に処理効率が低いため，秘匿関数計算を論理回路演算と算術演算に分けて全体の処理効率を上げる手法も提案されており[21)–24)], CIC48 方式も同様の方針により高速化を目指した方式である．

3. 準備：CIC48 方式, 情報理論的安全性

3.1 CIC48 方式

CIC48 方式は，計算対象の入力値を秘匿しつつ計算結果を求めるために，3 主体 X, Y, Z の協調計算を必要とする．なお対象の計算は処理効率を上げるために論理回路演算と算術演算に分けられているものとする．

入力値の秘匿処理は 3.1 節のとおり単純な秘密分散となる．復元は 2 主体の分散データの加算により実現され，単独の主体の分散データでは復元できない．ただし分散データを入出力とした加減算や定数倍演算は，既存方式[17),18)] 同様，各主体が単独で計算できる．乗算や論理回路演算については，既存方式[17),18)] 同様，各主体の協調計算が必要となる．更に提案方式は，2 進数変換や整数変換についても他の計算の単純な拡張として実現でき，これにより算術演算および論理回路演算の組合せ計算が可能となる．

本方式では，2 主体の取得情報を得ることはできないと仮定する．言い換えれば各主体は他の主体と結託しないものとする．また計算は $\mathbb{Z}/m\mathbb{Z}$ 上で行われるものとし，計算の効率化のため $m = 2^\ell$（ℓ は適当な整数）として話を進める．

[秘匿処理]

データ提供主体 P は $a \in \mathbb{Z}/m\mathbb{Z}$ を以下のように秘密分散し，X, Y, Z の入力とする．

X： $a_x = a_y + a_z$
Y： $\vec{a}_y = (\hat{a}, a_y)$
Z： $\vec{a}_z = (\hat{a}, a_z)$

ただし $a_y, a_z \in_R \mathbb{Z}/m\mathbb{Z}$, $\hat{a} = a - a_x$.

上記の秘匿処理はデータ提供主体が実行する．データ提供主体は X, Y, Z の何れかであっても良いし，外部の主体でも良い．手続きとしては乱数 a_y, a_z を生成してから a_x, $\hat{a}$ を計算して各主体に分散データを送信すれば良い．

a_x, $\vec{a}_y$, $\vec{a}_z$ は何れもその取り方から a と独立の乱数とみなせるため，X, Y, Z 単独では a を復元できない．

Vol.2010-CSEC-50 No.46
2010/7/2

[復元処理]

以下の処理を実行し, a を復元する.

(1) 以下の何れかを行い復元に必要なデータを計算依頼主体 C に送信する.

- X, Y: a_x, $\hat{a}$ をそれぞれ C に送信する.
- Y, Z: a_y, a_z をそれぞれ C に送信する.
- X, Z: a_x, $\hat{a}$ をそれぞれ C に送信する.

(2) C は次の 2 式の何れかに従い a を求める.

$a = a_x + \hat{a}$

$a = a_y + a_z + \hat{a}$

上記復元処理は関数計算の結果を得る段階であるため, 最終的な処理は計算依頼主体が行う. (2) における 2 式より a を正しく復元できることは $\hat{a} = a - a_x$, $a_x = a_y + a_z$ より明らであろう.

[加減算]

X, Y, Z は a, b の分散データ (a_x, b_x), $(\vec{a}_y, \vec{b}_y)$, $(\vec{a}_z, \vec{b}_z)$ から $c = a \pm b$ の分散データ c_x, $\vec{c}_y$, $\vec{c}_z$ を求める.

X : $c_x = a_x \pm b_x$

Y : $\vec{c}_y = (\hat{c}, c_y) = (\hat{a} \pm \hat{b}, a_y \pm b_y)$

Z : $\vec{c}_z = (\hat{c}, c_z) = (\hat{a} \pm \hat{b}, a_z \pm b_z)$

以下が成り立つことから, c_x, $\vec{c}_y$, $\vec{c}_z$ の何れか二つから c を復元できることが分かる.

$$c_x + \hat{c} = (a_x \pm b_x) + (\hat{a} \pm \hat{b}) = (a_x + \hat{a}) \pm (b_x + \hat{b}) = a \pm b$$
$$c_y + c_z = (a_y \pm b_y) + (a_z \pm b_z) = (a_y + a_z) \pm (b_y + b_z) = a_x \pm b_x = c_x$$

[定数倍演算]

X, Y, Z は a の分散データ a_x, $\vec{a}_y$, $\vec{a}_z$ および定数 $e \in \mathbb{Z}/m\mathbb{Z}$ から $c = ea$ の分散データ c_x, $\vec{c}_y$, $\vec{c}_z$ を求める.

X : $c_x = e(a_x \pm b_x)$

Y : $\vec{c}_y = (\hat{c}, c_y) = (e\hat{a}, ea_y)$

Z : $\vec{c}_z = (\hat{c}, c_z) = (e\hat{a}, ea_z)$

c_x, $\vec{c}_y$, $\vec{c}_z$ の何れか二つから c を復元できることは加減算同様に確認すれば良い.

[乗算]

X, Y, Z は a, b の分散データ (a_x, b_x), $(\vec{a}_y, \vec{b}_y)$, $(\vec{a}_z, \vec{b}_z)$ から $c = ab$ の分散データ c_x, $\vec{c}_y$, $\vec{c}_z$ を求める.

(1) X は以下を行う.

(a) $r_1, r_2, r_3, r_4, c_y \in_R \mathbb{Z}/m\mathbb{Z}$ を生成する.

(b) $c_x = a_x b_x - r_3 - r_4$ を計算して c の分散データとする.

(c) $c_z = c_x - c_y$ を計算する.

(d) Y, Z にそれぞれ (r_1, r_2, r_3, c_y), $(a_x - r_1, b_x - r_2, r_4, c_z)$ を送信する.

(2) Y, Z はそれぞれ以下を計算して共有する.

$$y = \hat{a}\hat{b} + \hat{a}r_2 + r_1\hat{b} + r_3$$
$$z = \hat{a}(b_x - r_2) + (a_x - r_1)\hat{b} + r_4$$

(3) Y, Z は $\hat{c} = y + z$ を計算してそれぞれ $\vec{c}_y = (\hat{c}, c_y)$, $\vec{c}_z = (\hat{c}, c_z)$ を c の分散データとする.

以下が成り立つことから, c_x, $\vec{c}_y$, $\vec{c}_z$ の何れか二つから c を復元できることが分かる.

$$\hat{c} = y + z = \hat{a}\hat{b} + \hat{a}b_x + a_x\hat{b} + r_3 + r_4$$
$$c_y + c_z = c_x = a_x b_x - r_3 - r_4$$
$$c_x + \hat{c} = (a_x + \hat{a})(b_x + \hat{b}) = ab$$

Vol.2010-CSEC-50 No.46
2010/7/2

[論理回路演算]

- 否定：X, Y, Z は $a \in \mathbb{Z}/2\mathbb{Z} \subseteq \mathbb{Z}/m\mathbb{Z}$ の分散データ a_x, $\vec{a}_y$, $\vec{a}_z$ から $\bar{a} = 1 - a$ の分散データ $\bar{a}_x$, $\vec{\bar{a}}_y$, $\vec{\bar{a}}_z$ を求める．
  X： $\bar{a}_x = -a_x$
  Y： $\vec{\bar{a}}_y = (\hat{\bar{a}}, \bar{a}_y) = (1 - \hat{a}, -a_y)$
  Z： $\vec{\bar{a}}_z = (\hat{\bar{a}}, \bar{a}_z) = (1 - \hat{a}, -a_z)$
- 論理積：X, Y, Z は $a, b \in \mathbb{Z}/2\mathbb{Z} \subseteq \mathbb{Z}/m\mathbb{Z}$ の分散データ (a_x, b_x), $(\vec{a}_y, \vec{b}_y)$, $(\vec{a}_z, \vec{b}_z)$ から $c = a \wedge b = ab$ の分散データ c_x, $\vec{c}_y$, $\vec{c}_z$ を求める．すなわち乗算を実行すれば良い．
- 論理和：X, Y, Z は $a, b \in \mathbb{Z}/2\mathbb{Z} \subseteq \mathbb{Z}/m\mathbb{Z}$ の分散データ (a_x, b_x), $(\vec{a}_y, \vec{b}_y)$, $(\vec{a}_z, \vec{b}_z)$ から $c = a \vee b = a + b - ab$ の分散データ c_x, $\vec{c}_y$, $\vec{c}_z$ を求める．すなわち加減算および乗算を実行すれば良い．
- 排他的論理和：X, Y, Z は $a, b \in \mathbb{Z}/2\mathbb{Z} \subseteq \mathbb{Z}/m\mathbb{Z}$ の分散データ (a_x, b_x), $(\vec{a}_y, \vec{b}_y)$, $(\vec{a}_z, \vec{b}_z)$ から $c = a \oplus b = a + b - 2ab$ の分散データ c_x, $\vec{c}_y$, $\vec{c}_z$ を求める．すなわち加減算，定数倍演算および乗算を実行すれば良い．

否定演算について，以下が成り立つことから，$\bar{a}_x$, $\vec{\bar{a}}_y$, $\vec{\bar{a}}_z$ の何れか二つから $\bar{a} = 1 - a$ を復元できることが分かる．

$$\hat{\bar{a}} = 1 - \hat{a} = 1 - (a - a_x)$$
$$\bar{a}_x + \hat{\bar{a}} = -a_x + (1 - (a - a_x)) = 1 - a$$
$$\bar{a}_y + \bar{a}_z = -a_y - a_z = -a_x = \bar{a}_x$$

[2 進変換処理]

X, Y, Z は $a \in \mathbb{Z}/m\mathbb{Z}$ の分散データ a_x, $\vec{a}_y$, $\vec{a}_z$ から $a[i] \in \mathbb{Z}/2\mathbb{Z} \subseteq \mathbb{Z}/m\mathbb{Z}$ $(i = 1, \ldots, \ell)$ の分散データ $a[i]_x$, $\vec{a}[i]_y$, $\vec{a}[i]_z$ を求める．ここで $a[i]$ は a の下位 i 番目のビットとする．

(1) X, Y はそれぞれ $a_x[i]$, $\hat{a}[i]$ $(i = 1, \ldots, \ell)$ の秘匿処理を実行する．

(2) X, Y, Z は式 (1) および以下の全加算回路演算式に基づき $a_x[i]$, $\hat{a}[i]$ の分散データから $a[i]$ の分散データを求める．

$$c_1 = 0$$
$$d_i = a_x[i] + \hat{a}[i] - 2a_x[i]\hat{a}[i]$$
$$\text{(sum) } a[i] = a_x[i] \oplus \hat{a}[i] \oplus c_i = d_i + c_i - 2d_ic_i \quad (1)$$
$$\text{(carry out) } c_{i+1} = (a_x[i] \wedge \hat{a}[i]) \vee (a_x[i] \wedge c_i) \vee (\hat{a}[i] \wedge c_i)$$
$$= a_x[i]\hat{a}[i] + d_ic_i - a_x[i]\hat{a}[i]d_ic_i \quad (2)$$

式 (1), (2) は単純な $a_x + \hat{a}(= a)$ の論理式であり，$a_x[i]\hat{a}[i]$, d_ic_i, $(a_x[i]\hat{a}[i])(d_ic_i)$ の乗算を実行する．全体では加減算および乗算がともに $3\ell - 2$ 回，そして定数倍が $2\ell - 1$ 回となる．乗算は並列化できないため，通信回数は $O(\ell)$ となる．なお上記手続き (2) の全加算回路演算を文献[22] の Protocol $[d]_B$ (Section 6.2) に置き換えれば，通信回数を $O(1)$ とできるが，乗算の回数が $55\ell \log_2 \ell$ となる．

[整数変換処理]

X, Y, Z は $a[i] \in \mathbb{Z}/2\mathbb{Z} \subseteq \mathbb{Z}/m\mathbb{Z}$ $(i = 1, \ldots, \ell)$ の分散データ $a[i]_x$, $\vec{a}[i]_y$, $\vec{a}[i]_z$ から $a \in \mathbb{Z}/m\mathbb{Z}$ の分散データ a_x, $\vec{a}_y$, $\vec{a}_z$ を求める．$a = \sum_{i=1}^{\ell} 2^{i-1}a[i] \bmod m$ が成り立つことから，加算および定数倍演算を実行すれば良い．

3.2 情報理論的安全性

情報理論的安全性は, 計算量的安全性が攻撃者の計算能力を多項式時間に制限するのに対し, 考え得る最も強力な攻撃者, すなわち任意の関数を計算可能であるような攻撃者に対する安全性である. 特に攻撃者にとって秘密のデータの推定分布がある関数による開示データを見ても全く変化しないとき, その関数は**完全秘匿性** (*perfec secrecy*) を持つという[25]. 例えば暗号化アルゴリズムであれば秘密は平文すなわち入力であり, 開示データは暗号文すな

わち出力である.

定義 3.1. 集合 $\mathcal{S}$ 上の確率変数 S と独立であるような, $\mathcal{S}$ から集合 $\mathcal{D}$ への確率的関数 F が完全秘匿性を持つとは, 任意の実際の秘密のデータ $s \in \mathcal{S}$ と出力 $d \in \mathcal{D}$ に対して,

$$\Pr(S = s|F(S) = d) = \Pr(S = s)$$

となることを言う.

上記で S が秘密のデータであり $F(S)$ が開示されるデータである.

一方秘匿関数計算を含む, マルチパーティ計算の安全性の定義については 26), 27) などで解説がされており, 計算量的安全性に関して "攻撃者の観測を既知のデータのみから多項式時間で再現できるなら安全" という定義がされている. この定義から "多項式時間で" を除けば情報理論的安全性の定義と見ることができる. さらに計算量的識別不能性を確率的等価性と置き換えれば, マルチパーティプロトコルにおける完全秘匿性の定義と見ることができる.

定義 3.2. j 番目のパーティに関するデータが以下であるような n パーティプロトコルを考える.

- 入力 : x_j
- 出力 : $f_j(x_1, \ldots, x_n)$,
- 出力を除く受信データ : $\mathrm{VIEW}_j(x_1, \ldots, x_n)$

このプロトコルが *semi-honest* モデルで完全秘匿性を持つとは, 以下のような確率的関数 S が存在することである.

任意の入力 $\vec{x} = \{x_1, \ldots, x_n\}$, $I = \{i_1, \ldots, i_{|I|}\} \subseteq \{1, \ldots, n\}$ に対して
$S(I, \vec{x_I}, f_I(\vec{x}))) = \mathrm{VIEW}_I(\vec{x})$

ただし $\vec{x_I} = \{x_{i_1}, \ldots, x_{i_{|I|}}\}, f_I = (f_{i_1}, \ldots, f_{i_{|I|}}), \mathrm{VIEW}_I = (\mathrm{VIEW}_{i_1}, \ldots, \mathrm{VIEW}_{i_{|I|}})$ である.

上記定義の中でパーティの集合 I が 2 パーティ以上の集合の場合は当該パーティが結託したことを表現している. このため本定義の安全性は, パーティ同士がどのように結託しようとも安全であることを保証する.

4. 本稿の主題:CIC48 方式の安全性

CIC48 方式は主体 X, Y, Z による 3 パーティプロトコルである. しかし今後場合によっては X, Y, Z はサーバで, エンドユーザがデータ提供主体として秘匿データをサーバに預託, そして計算結果はまた別のサービス事業者が得る, といったことが想定される. このようなことを念頭に置くと, 厳密にはエンドユーザすなわちデータ提供主体が $\{P_1, \ldots, P_k\}$ の k パーティ, サーバすなわち計算主体が X, Y, Z の 3 パーティ, サービス事業者すなわち計算依頼主体が $\{C_1, \ldots, C_{k'}\}$ の k' パーティとするのが最も一般性が高く妥当である.

本稿では定義 3.2 の定義に従い安全性を証明する. ただ, 定義 3.2 の定義は結託に対する耐性をも保証するが, CIC48 方式は計算主体の結託に対する安全性をスコープとしない. そのため本稿では定義 3.2 の条件をパーティ集合 I 毎に分離した以下の定義をおき, 計算主体の結託を含まないようなパーティ集合についての安全性を議論する.

定義 4.1. 定義 3.2 と同様のプロトコルを考え, $I = \{i_1, \ldots, i_t\} \subseteq \{1, \ldots, m\}$ とする. このプロトコルが *semi-honest* モデルでパーティ集合 I に対して完全秘匿性を持つとは, 以下のような確率的関数 S が存在することである.

任意の入力 $\vec{x} = \{x_1, \ldots, x_m\}$ に対して $S(I, \vec{x_I}, f_I(\vec{x}))) = \mathrm{VIEW}_I(\vec{x})$

特に以下は本稿の証明で用いる, 定義 4.1 の条件の十分条件である.

命題 4.1. 定義 3.2 と同様のプロトコルを考え, $I \subseteq \{1, \ldots, n\}$ とする. このとき, $\mathrm{VIEW}_I(\vec{x})$ が一様乱数ならばプロトコルは *semi-honest* モデルでパーティ集合 I に対して完全秘匿性を持つ.

Proof. 定義中, S を常に一様乱数を出力する関数とすればよい. □

では以下に CIC48 方式の安全性を評価していく. CIC48 方式は単一のプロトコルではなく, 3.1 節に示した各プロトコルの複合から成るプロトコルの集合である. そのため, 全ての複合したプロトコルが安全であることを示す必要がある. 以降, 3.1 節に示したプロトコル群を基本プロトコル, 並列及び直列の接続によりそれらを複合したプロトコルを複合プロトコルと呼ぶ. すると本稿の示すべき命題は以下となる.

Vol.2010-CSEC-50 No.46
2010/7/2

定理 4.1. CIC48 方式による任意の複合プロトコルは, X, Y, Z の結託を許さない *semi-honest* モデルにおいて完全秘匿性を持つ.

Proof. まずデータ提供主体が関与する基本プロトコルは入力の秘匿処理のみである. 秘匿処理ではデータを送信するのみであって受信データが存在しないため, いかなる複合プロトコルに対してもデータ提供主体の VIEW は空である. また計算依頼主体は復元処理のみを行うため受信データは計算結果のみであり, やはり VIEW は空である. これらからデータ提供主体, 計算依頼主体の存在は安全性に影響を与えないことが分かるため, これらの VIEW には以降言及しない.

X, Y, Z の 3 パーティの VIEW に関しては, まず各基本プロトコルの VIEW を考え, そこから帰納的に VIEW を定義する. 基本プロトコルのうち, 加減算, 定数倍演算はデータ送受信を伴わないため, VIEW は空である. また論理回路演算, 2 進変換処理, 整数変換処理は秘匿処理, 加減算, 定数倍演算, 乗算から構成されるため後者の複合プロトコルであると見なすことができる. そして復元処理の計算主体の VIEW は空である. これらから, 結局秘匿処理, 乗算の 2 プロトコルのみが実際考慮する必要のある対象である. 上記 2 プロトコルの VIEW を表す確率的関数をそれぞれ V^σ, V^μ とすると, それぞれ以下のようになる.

$$
\begin{aligned}
V^\sigma : \mathbb{Z}/m\mathbb{Z} &\to [\![\mathbb{Z}/m\mathbb{Z}]\!] \times [\![(\mathbb{Z}/m\mathbb{Z})^2]\!] \times [\![(\mathbb{Z}/m\mathbb{Z})^2]\!] \\
a &\mapsto (R_x, \\
&\quad (a - R_x, R_y), \\
&\quad (a - R_x, R_x - R_y)) \\
V^\mu : \mathbb{Z}/m\mathbb{Z}^4 &\to \{\bot\} \times [\![(\mathbb{Z}/m\mathbb{Z})^5]\!] \times [\![(\mathbb{Z}/m\mathbb{Z})^5]\!] \\
(a_x, \hat{a}, b_x, \hat{b}) &\mapsto (\bot, \\
&\quad (R_1, R_2, R_3, \hat{a}(b_x - R_2) + (a_x - R_1)\hat{b} + R_4, R_y), \\
&\quad (a_x - R_1, b_x - R_2, \hat{a}\hat{b} + \hat{a}R_2 + R_1\hat{b} + R_3, R_4, a_x b_x - R_3 - R_4 - R_y))
\end{aligned}
$$

ただし $R_x, R_y, R_z, R_1, R_2, R_3, R_4$ は互いに独立な一様乱数, $\bot$ は空データ, 集合 Ω に対して $[\![\Omega]\!]$ は Ω 上の確率変数の集合を表す. 両者共, 関数値の第 1 成分から順に X, Y, Z の VIEW である.

命題 4.1 より複合プロトコルが一様乱数であることを示せばよいのだが, その前にまず V^σ, V^μ が一様乱数であることを示し, その後複合プロトコルの VIEW が一様乱数であることを示す.

補題 4.1. 任意の $a, a_x, \hat{a}, b_x, \hat{b} \in \mathbb{Z}/m\mathbb{Z}$ に対して,
$V_X^\sigma(a), V_Y^\sigma(a), V_Z^\sigma(a), V_X^\mu(a_x, \hat{a}, b_x, \hat{b}), V_Y^\mu(a_x, \hat{a}, b_x, \hat{b}), V_Z^\mu(a_x, \hat{a}, b_x, \hat{b})$
は一様乱数である.

Proof. (補題)
それぞれ一様乱数性を調べればよい. $R_x, R_y, R_z, R_1, R_2, R_3, R_4$ が互いに独立な一様乱数であることと, $V_Y^\sigma, V_Z^\sigma, V_Y^\mu, V_Z^\mu$ の値域がそれぞれ m^2, m^2, m^5, m^5 であることに注意すれば, $w_1, w_2, w_3, w_4, w_5 \in \mathbb{Z}/m\mathbb{Z}$ を任意として以下の通りいずれも一様乱数であることが分かる.

V_X^σ : R_xが一様乱数より.

$$
\begin{aligned}
V_Y^\sigma &: \Pr((a - R_x, R_y) = (w_1, w_2)) = \Pr((R_x, R_y) = (a - w_1, w_2)) = \frac{1}{m^2} \\
V_Z^\sigma &: \Pr((a - R_x, R_x - R_y) = (w_1, w_2)) = \Pr((R_x, R_x - R_y) = (a - w_1, w_2)) \\
&= \Pr(R_x = a - w_1)\Pr(R_x - R_y = w_2 | R_x = a - w_1) \\
&= \frac{1}{m}\Pr(a - w_1 - R_y = w_2 | R_x = a - w_1) = \frac{1}{m}\Pr(a - w_1 - R_y = w_2) \\
&= \frac{1}{m}\Pr(R_y = a - w_1 - w_2) = \frac{1}{m^2} \\
V_X^\mu &: \text{値域が一点集合より一様乱数.} \\
V_Y^\mu &: \Pr((R_1, R_2, R_3, \hat{a}(b_x - R_2) + (a_x - R_1)\hat{b} + R_4, R_y) = (w_1, w_2, w_3, w_4, w_5)) \\
&= \frac{1}{m^4}\Pr(R_4 = w_4 - \hat{a}(b_x - w_2) + (a_x - w_1)\hat{b}) = \frac{1}{m^5} \\
V_Z^\mu &: \Pr((a_x - R_1, b_x - R_2, \hat{a}\hat{b} + \hat{a}R_2 + R_1\hat{b} + R_3, R_4, a_x b_x - R_3 - R_4 - R_y) \\
&\qquad = (w_1, w_2, w_3, w_4, w_5)) \\
&= \frac{1}{m^3}\Pr((\hat{a}\hat{b} + \hat{a}w_2 + w_1\hat{b} + R_3, a_x b_x - R_3 - w_4 - R_y) = (w_3, w_5)) \\
&= \frac{1}{m^4}\Pr(a_x b_x - w_3 - w_4 - R_y = w_5) = \frac{1}{m^5}
\end{aligned}
$$

□

補題により, 秘匿処理及び乗算処理単体が安全であることが示された. 次にこれらが複合された場合の一様乱数性について考える. CIC48 方式においては計算主体は入力をもたないため, 複合プロトコルはある時点までの VIEW 及び定数を入力としてさらに基本プロトコルを行うことで生成される. このとき VIEW はその時点の VIEW にそのまま次の処理の VIEW が直積として追加される.

Vol.2010-CSEC-50 No.46
2010/7/2

まずパーティX の VIEW について考える．X に関しては, VIEW は秘匿処理に関するもののみであり, データ提供主体により与えられた入力が k 個とすれば, 複合プロトコルにおける VIEW_X は秘匿処理の VIEW の直積 $(R_{x_1}, \ldots, R_{x_k})$ である．各 $R_{x_1}, \ldots, R_{x_k}$ は互いに独立な一様乱数であり, VIEW_X も一様乱数であることがわかる．

パーティY, Z に関しては, 秘匿処理及び乗算処理の実行回数に関する帰納法で VIEW が一様乱数であることを示す．

(1) 実行回数が 0 回のとき, パーティX と同様に $\mathrm{VIEW}_Y, \mathrm{VIEW}_Z$ は一様乱数である．

(2) 実行回数が i 回以下の任意の複合プロトコルに対する $\mathrm{VIEW}_Y, \mathrm{VIEW}_Z$ が一様乱数であると仮定する．実行回数が $i+1$ の複合プロトコルを考えると, 実行回数 i 回のある複合プロトコルが存在して, その VIEW を $\mathrm{VIEW}_Y^i, \mathrm{VIEW}_Z^i$ とおけば, 両者は一様乱数である．また入力の確率変数を A, i 回の実行及び秘匿処理に使用された乱数をまとめて R とおけば, 乱数生成以外の処理は決定的であるから, ある決定的関数 v_Y, v_Z が存在して $v_Y(A,R) = \mathrm{VIEW}_Y^i, v_Z(A,R) = \mathrm{VIEW}_Z^i$ であり, $i+1$ 回目の実行の入力に関しても同様に A, R の決定的関数で表すことができる．よって実行後の VIEW を $\mathrm{VIEW}_Y^{i+1}, \mathrm{VIEW}_Z^{i+1}$ とおけば, 両者は次のように表される．

(I) $i+1$ 回目が秘匿処理の場合
秘匿処理の入力を $a(A,R)$ とおけば

$$\mathrm{VIEW}_Y^{i+1} = (v_Y(A,R), V_Y^{\sigma}(a(A,R)))$$
$$\mathrm{VIEW}_Z^{i+1} = (v_Z(A,R), V_Z^{\sigma}(a(A,R)))$$

(II) $i+1$ 回目が乗算処理の場合
乗算処理の入力を $a_x(A,R), \hat{a}(A,R), b_x(A,R), \hat{b}(A,R)$ とおけば

$$\mathrm{VIEW}_Y^{i+1} = (v_Y(A,R), V_Y^{\mu}(a_x(A,R), \hat{a}(A,R), b_x(A,R), \hat{b}(A,R)))$$
$$\mathrm{VIEW}_Z^{i+1} = (v_Z(A,R), V_Z^{\mu}(a_x(A,R), \hat{a}(A,R), b_x(A,R), \hat{b}(A,R)))$$

次に $\mathrm{VIEW}_Y^{i+1} = (v_Y(A,R), V_Y^{\sigma}(a(A,R)))$ が一様乱数であることを示す．なお，他の 3 つの場合も証明は全く同様である．関数 v_Y の値域集合を $\mathcal{D}_v$ とおけば，任意の $d_1 \in \mathcal{D}_v, d_2 \in (\mathbb{Z}/m\mathbb{Z})^2$ に対して以下のように $\Pr(\mathrm{VIEW}_Y^{i+1} = (d_1, d_2))$ が等しいことが分かり, 一様乱数性が示される．

$$\begin{aligned}
\Pr(\mathrm{VIEW}_Y^{i+1} = (d_1,d_2)) &= \Pr((v(A,R), V_Y^{\sigma}(a(A,R))) = (d_1,d_2)) \\
&= \Pr(v(A,R) = d_1)\Pr(V_Y^{\sigma}(a(A,R)) = d_2 | v(A,R) = d_1) \\
&= \frac{1}{|\mathcal{D}_v|}\Pr(V_Y^{\sigma}(a(A,R)) = d_2 | (A,R) \in v^{-1}(\{d_1\})) \\
&= \frac{1}{|\mathcal{D}_v|}\sum_{r \in v^{-1}(\{d_1\})} \Pr(V_Y^{\sigma}(a(A,R)) = d_2 | (A,R) = r) \\
&= \frac{1}{|\mathcal{D}_v|}\sum_{r \in v^{-1}(\{d_1\})} \Pr(V_Y^{\sigma}(a(r)) = d_2) \\
&= \frac{1}{m^2|\mathcal{D}_v|}
\end{aligned}$$

よって帰納法の仮定により任意の有限回の秘匿処理，乗算処理の実行について $\mathrm{VIEW}_Y, \mathrm{VIEW}_Z$ は一様乱数である．すなわち，CIC48 方式に属する任意の複合プロトコルは X, Y, Z の結託を許さない semi-honest モデルにおいて, 完全秘匿性を持つ． □

5. おわりに

本稿では高効率な 3 パーティ秘匿関数計算方式である, CIC48 方式に関してその安全性を議論し, 当方式が結託を許さない semi-honest モデルにおいて完全秘匿性を持つことを示した．当方式は汎用的な計算を実現する論理回路と共に, 最も頻繁に用いられる演算である算術演算の高効率化のために加減算, 乗算を基本演算として持つ, 汎用性と高速性を併せ持つ方式であり, さらに完全秘匿性を持つことを示したことでさらに安全性も高いことが示されたこととなる．

今後の課題としてはさらに, 当方式を実装することでその高速性を実証すること, また発展として当方式と同時に提案した malicious モデル対応方式の安全性を評価することなどが挙げられる．

参 考 文 献

1) 健康情報活用基盤構築のための標準化及び実証事業, `https://microsite.accenture.com/meti/Pages/default.aspx`.
2) 情報大航海プロジェクト, `http://www.igvpj.jp/index/`.
3) 「地理空間情報サービス産業の将来ビジョン」及び「G 空間プロジェクト」の公表について（METI/経済産業省）,

http://www.meti.go.jp/press/20080703007/20080703007.html.
4) Geographic Privacy-aware Knowledge Discovery and Delivery, http://www.geopkdd.eu/.
5) Electronic Health Information Laboratory – KnowledgeBase, http://www.ehealthinformation.ca/knowledgebase/.
6) A. C. Yao, Protocols for secure computations, FOCS '82, pp. 160–164, IEEE Press, 1982.
7) Y. Lindell and B. Pinkas, Privacy preserving data mining, CRYPTO 2000, LNCS 1880, pp. 36–54, Springer-Verlag, 2000.
8) J. Vaidya, C.W. Clifton, and Y.M. Zhu, Privacy Preserving Data Mining, ISBN 0-387-25886-8, Springer-Verlag, Nov. 2005.
9) C.C. Aggarwal and P.S. Yu, Privacy-Preserving Data Mining: Models and Algorithms, ISBN 978-0-387-70991-8, Springer-Verlag, Jul. 2009.
10) 千田 浩司, 五十嵐 大, 高橋 克巳. 効率的な 3 パーティ秘匿関数計算の提案とその運用モデルの考察. CSEC48, 2010.
11) A. C. Yao, How to generate and exchange secrets, FOCS '86, pp. 162–167, IEEE Press, 1986.
12) O. Goldreich, S. Micali, and A. Wigderson, How to play any mental game, or a completeness theorem for protocols with honest majority, STOC '87, pp. 218–229, ACM Press, 1987.
13) M. Jakobsson and A. Juels, Mix and match: secure function evaluation via ciphertexts, ASIACRYPT 2000, LNCS 1976, pp. 162–177, Springer-Verlag, 2000.
14) C. Gentry, Fully homomorphic encryption using ideal lattices, STOC '09, pp. 169–178, ACM Press, 2009.
15) B. Pinkas, T. Schneider, N.P. Smart, and S.C. Williams, Secure two-party computation is practical, ASIACRYPT 2009, LNCS 5912, pp. 250–267, Springer-Verlag, 2009.
16) 柴田 賢介，千田 浩司，五十嵐 大，山本 太郎，高橋 克巳，表計算ソフトをフロントエンドとした委託型 2 パーティ秘匿回路計算システム，CSS2009, 2009.
17) M. Ben-Or, S. Goldwasser, and A. Wigderson, Completeness theorems for non-cryptographic fault-tolerant distributed computation, STOC '88, pp. 1–10, ACM Press, 1988.
18) D. Chaum, C. Crepeau, and I. Damgård, Multiparty unconditionally secure protocols, STOC '88, pp. 11–19, ACM Press, 1988.
19) R. Cramer, I. Damgård, and J. B. Nielsen, Multiparty computation from threshold homomorphic encryption, EUROCRYPT 2001, LNCS 2045, pp. 280–300, Springer-Verlag, 2001.
20) B. Schoenmakers and P. Tuyls, Practical two-party computation based on the conditional gate, ASIACRYPT 2004, LNCS 3329, pp. 119–136, Springer-Verlag, 2004.
21) J. Algesheimer, J. Camenisch, and V. Shoup, Efficient computation modulo a shared secret with application to the generation of shared safe-prime products, CRYPTO 2002, LNCS 2442, pp. 417–432, Springer-Verlag, 2002.
22) I. Damgård, M. Fitzi, E. Kiltz, J. B. Nielsen, and T. Toft, Unconditionally secure constant-rounds multi-party computation for equality, comparison bits and exponentiation, TCC 2006, LNCS 3876, pp. 285–304, Springer-Verlag, 2006.
23) T. Nishide and K. Ohta, Multiparty computation for interval, equality, and comparison without bit-decomposition protocol, PKC 2007, LNCS 4450, pp. 343–360, Springer-Verlag, 2007.
24) B. Schoenmakers and P. Tuyls, Efficient binary conversion for Paillier encrypted values, EUROCRYPT 2006, LNCS 4004, pp. 522–537, Springer-Verlag, 2006.
25) C. E. Shannon, Communication theory of secrecy system, Bell System Technial Journal, Vol.28-4, pp.656-715, Oct. 1949.
26) O. Goldreich, Secure Multi-Party Computation (Final (incomplete) Draft, Version 1.4), http://www.wisdom.weizmann.ac.il/~oded/pp.html 2002.
27) O. Goldreich, Foundations of Cryptography, Vol.2, Cambridge University Press, 2004.