

秘匿回路計算の高効率化と 機密情報の安全な活用について*1

千田 浩 司^{†1} 五十嵐 大^{†1} 柴田 賢 介^{†1}
山本 太郎^{†1} 高橋 克 巳^{†1}

入力データや演算ロジックを秘匿しつつ各種情報処理を可能とする技術の実現可能性が 1982 年に Yao によって提起されたが、実用上は非現実的な処理時間を要するためもっぱら理論研究のみにとどまっていた。しかしながら近年では、アルゴリズム改良や計算・通信環境の急速な発達に加え、医療分野やサービス分野等での個人のプライバシーに関わる情報の安全な活用や、クラウドコンピューティングにおける機密情報保護等の社会的ニーズの高まりを背景に、当該技術に対する実装報告や実用化の動きも見られるようになった。本論文では、当該技術のうち特に情報処理の種別を限定せず汎用的に適用可能な秘匿回路計算 (Secure Circuit Evaluation) 技術に着目し、従来のアプローチを概観した後、より効率的に処理可能、かつ運用上の利点が見込める委託型 2 パーティ秘匿回路計算を提案する。また実装により提案方式のパフォーマンスを明らかにするとともに、実用上の価値や課題を探るため実証実験を行った結果について報告する。さらに、個人のプライバシーに関わる情報の安全な活用や、クラウドコンピューティングにおける機密情報保護の実現に向け、技術的視点から考察する。

A Research on Efficient Secure Circuit Evaluation and Its Application to Secure Utilization of Sensitive Information

KOJI CHIDA,^{†1} DAI IKARASHI,^{†1} KENSUKE SHIBATA,^{†1}
TARO YAMAMOTO^{†1} and KATSUMI TAKAHASHI^{†1}

A cryptographic technology concept that achieves various information processing keeping input data and/or an operation logic secret was proposed by Yao in 1982; however, it has entirely been stayed only in the theory research due to a heavy processing time. Recently, however, social needs for utilizing personal information safely in the fields of medicine and services etc. and for the *cloud computing* security are increasing with rapid development of ICT (information and communication technology) environments. In this paper, we focus on *secure circuit evaluation* as a solution for the Yao's concept and propose

a delegation-based 2-party secure circuit evaluation. Moreover, we report on an empirical result of the proposed scheme to clarify the performance and to consider the potentiality on practical use, in particular, for safe uses of personal information and cloud computing security.

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、個人情報保護やプライバシーの観点から、医療分野やサービス分野等において、個人に関する情報（以後、これをパーソナル情報と呼ぶ）の利活用について国内外で法制度・ガイドラインや標準化の整備とあわせた技術的対策が検討されている^{4)–8)}。また、インターネットに接続さえすれば即座に各種のサービスが利用できるインフラが整備されつつあり、最近ではグローバルに拡散した計算資源を用いてサービス向上を図るクラウドコンピューティングが注目を集めているが、一般に外部に情報を預けることから、情報保護に対する問題解決は重要なセキュリティ課題である^{9)–11)}。しかし情報を利用するためには一般に当該情報へのアクセスを許可する必要があるため、不正アクセス、コンピュータウイルス等のマルウェア、そしてアクセス権限を持つ正規者の内部不正といった様々なリスクが生じ、その根本的な解決は容易ではない。実際、上記に起因する情報漏洩の事件が後を絶たない。

それでは情報の保護と利用を両立させるためにはどのような技術的対策が有効となるだろうか。パーソナル情報の利活用についていえば、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、パーソナル情報が個人と結び付くことのないように情報を加工する手段全般を指す。すなわち、情報の匿名化により情報漏洩時の被害を低減させる考え方である。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、パーソナル情報が個人と結び付かないことの保証が容易ではない。その理由の 1 つとして、パーソナル情報と個人の結び付けが、あらかじめ備えている知識や情報に大きく左右されることがあげられる。たとえば、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

^{†1} NTT 情報流通プラットフォーム研究所

NTT Information Platform Laboratories

*1 本論文は文献 1)–3) の内容を含む。

定の個人の購買履歴がすべて紐付いてしまう可能性があり、これはプライバシーの観点から望ましくない。このように、匿名化の強度や十分性（これは個人個人の主観的な判断によることも大きい）が匿名化技術の実用上の大きな課題である。

一方、匿名化とは別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情報処理を実現する秘匿関数計算 (Secure Function Evaluation)¹²⁾ 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ー保護データマイニング (Privacy-Preserving Data Mining) 技術は、Lindell らによる研究成果¹³⁾ 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^{14),15)}。しかしながら秘匿関数計算は一般に通常の情報処理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。なお本論文では、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ー保護データマイニングと呼ぶことにする^{*1}。

秘匿関数計算は、クラウドコンピューティングにおける情報保護技術としても有効である。クラウドコンピューティングにおいては、拡散した計算資源を用いることから運用による徹底したセキュリティ対策を実現することは容易ではない。また、組織がネットワークを通じて機密情報を外部に預ける行為自体が組織のセキュリティポリシーに反することも十分に考えられ、場合によってはクラウドコンピューティングの普及阻害要因となる可能性もある。しかし預けるデータがつねに秘匿されている状態であれば、情報漏洩のリスクは大幅に低下することが期待できる。

本論文では、秘匿関数計算の中でも特に情報処理の種別を限定せず汎用的に適用可能な秘匿回路計算 (Secure Circuit Evaluation) 技術に着目し、従来の手法と比べ、より効率的に処理可能、かつ運用上の利点が見込める秘匿回路計算方式を提案する。提案方式は、Naor らによって提案された委託型 2 パーティ秘匿回路計算¹⁶⁾ のモデル (図 1) を基本とし、Naor らの方式において情報処理の種別によっては支配的な計算になると考えられる、紛失通信 (Oblivious Transfer) プロトコル¹⁷⁾ を不要としたことが最大の特徴である。委託型 2 パーティ秘匿回路計算は、外部からデータを受信したある 2 つの計算主体 (図 1 では Proxy および Server が計算主体であり、Proxy が情報 a_i ($i = 1, \dots, N$) を秘匿したデータ a_i^* を受信) が協調して計算を行うモデルであり、2 つの計算主体の内部情報を得ない限り元の情報

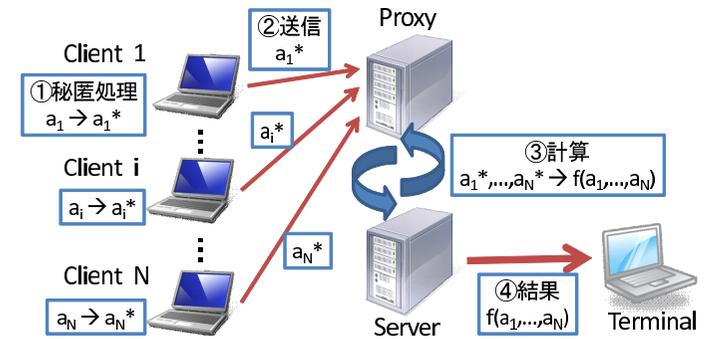


図 1 委託型 2 パーティ秘匿回路計算のモデル

Fig. 1 A delegation-based 2-party secure circuit evaluation model.

(図 1 では各 Client が所有する情報 a_i) の秘匿性のある種の仮定の下で保証する。これにより片方の計算主体の内部情報が故意または過失等により漏洩した場合でも、もう片方の計算主体の内部情報を得ない限り元の情報の復元は困難であるため、通常よりも非常に高いレベルで情報保護を実現できる。なお計算主体による計算結果 (図 1 では Terminal が取得するデータ $f(a_1, \dots, a_N)$) は、元の情報と比べ漏洩時の被害が十分小さいと仮定する。いい換えれば、計算結果から元の情報の復元は困難またはごく一部であるとする。処理内容によっては計算結果にも大きなリスクが存在しうるが、検索等のデータベース操作や統計処理においては、本仮定は十分妥当であると考えられる。

本論文ではまた、実装により提案方式のパフォーマンスを明らかにするとともに、実用上の価値や課題を探るため実験を行った結果について報告する。実用上は非現実的な処理時間を要するとおもわれていた秘匿関数計算は、アルゴリズム改良や計算・通信環境の急速な発達に加え、パーソナル情報の安全な活用や、クラウドコンピューティングにおける機密情報保護等、当該技術に対する社会的ニーズの高まりを背景に、近年では実装報告や実用化の動きも見られるようになった¹⁸⁾⁻²⁴⁾。秘匿関数計算は、限定された演算のみを行うことが可能な特化型の手法と、任意の論理回路を計算できる汎用型の手法、すなわち秘匿回路計算が存在するが、文献 22)-24) で用いられている手法は特化型であり、文献 18)-21) で用いられている手法は汎用型である。特化型は演算を限定する代わりに汎用型よりも一般に計算コストが低い。6 章で説明する行動分析実験は秘匿回路計算を利用した実験であり、汎用型の秘

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

密関数計算も実用的な性能を持ち始めていることを示すものである*1.

以降、2章で関連研究を紹介し、3章で提案方式について説明する。次に提案方式の安全性評価および実装評価についてそれぞれ4、5章で述べ、6章で実験報告を行う。7章では秘匿関数計算の応用として、パーソナル情報の安全な活用や、クラウドコンピューティングにおける機密情報保護の実現に向け、技術的視点から考察する。最後に8章でまとめと今後の課題について述べる。

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な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1986年にYaoによって提案された²⁵⁾。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、紛失通信プロトコルを利用した方式²⁶⁾や、MIX-net²⁷⁾を利用した方式²⁸⁾等が提案されている。なお秘匿回路計算は任意の論理回路演算の実行を可能とする秘匿関数計算を指し、非常に高い汎用性を持つ。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完全準同型暗号²⁹⁾が注目を集めているが、処理効率が悪く実用レベルには至っていないと考えられる。一方Yaoの方式に基づく2パーティ秘匿回路計算に関する実装報告も見られるようになり^{18),19)}、実用に向けた動きが加速しつつある。

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^{30),31)}や準同型暗号に基づく方式^{32),33)}が知られている。なお秘匿回路計算は一般に処理効率が悪いので、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いる³⁴⁾⁻³⁷⁾。

2パーティプロトコルと3主体以上のマルチパーティプロトコルは、セキュリティや運用面において以下の特徴の違いがあると考えられる。

- セキュリティ：論理回路の実行や秘匿処理されたデータの復元について、より多くの主

体の協調計算が必要となるマルチパーティプロトコルがより優れる。

- 運用：各主体について運用管理をとるため、運用の負担については一般に2パーティプロトコルの方が優れる。

汎用型のマルチパーティプロトコルはいくつか開発が進められているが⁴⁴⁾⁻⁴⁷⁾、比較的運用に優れる汎用型の2パーティプロトコルの例は筆者らが知る限り少ない⁴⁸⁾。本論文では運用面を重視した2パーティプロトコル、特に入力値を提供する主体の数は任意とできる委託型2パーティプロトコルに着目する。

以下では、提案手法の基本モデルである委託型2パーティ秘匿回路計算の先行研究¹⁶⁾について、その要素技術や関連技術を含め説明する。

2.1 紛失通信

紛失通信とは、1981年にRabinによって提案された暗号プロトコルであり³⁸⁾、 $1/2$ の確率で送信メッセージが受信者に届き、送信者は受信の成否が分からないという特徴を持つ。後に紛失通信は k -out-of- n 紛失通信と呼ばれる、 n 個のメッセージのうち受信者は指定した k 個のメッセージを送信者に知られることなく受信できる暗号プロトコルに発展し¹⁷⁾、暗号プロトコル設計の要素技術として重要な役割を担っている。以下では文献16)の記述に基づき、ElGamal暗号を用いて実現される中継型1-out-of-2紛失通信(Proxy 1-out-of-2 Oblivious Transfer)について紹介する。

q を大きな素数、 G_q を位数 q の巡回群とし、 g を G_q の生成元とする。 (q, G_q, g) は公開情報とし、ElGamal暗号の秘密鍵および公開鍵をそれぞれ $SK = u \in \mathbb{Z}/q\mathbb{Z}$ 、 $PK = y = g^u \in G_q$ とする。このとき、平文 $m \in G_q$ の暗号文は $E_{PK}(m, r) = (g^r, my^r) \in G_q^2$ となる($r \in_R \mathbb{Z}/q\mathbb{Z}$)。以降、記号の簡略化のため、乱数 r の記号を省略して $E_{PK}(m)$ と表記する。復号は、暗号文 (g^r, my^r) および秘密鍵 u を用いて、 $m = (my^r)/(g^r)^u$ を計算すればよい。なお $\mathbb{Z}/q\mathbb{Z}$ から G_q への同型写像 ϕ およびその逆写像 ϕ^{-1} が効率良く計算可能であるとすると、また、ある集合 G から G_q へ写すエラー訂正符号 $C: G \rightarrow G_q$ が存在し、 C および C^{-1} は ϕ 同様効率良く計算可能であるとすると、すなわち平文空間を $G_q \cup \mathbb{Z}/q\mathbb{Z} \cup G$ とする。

中継型1-out-of-2紛失通信はClient, Proxy, およびServerによる以下の手続きによって実現される。なお文献16)ではそれぞれSender, Chooser, およびProxyに対応する。本論文とはProxyの役割が異なることに注意されたい。

共通入力： (q, G_q, g, ϕ, C, G) 、 z (z は適当な G_q の元)

Clientの入力： $\sigma \in \{0, 1\}$ 、 PK_S (Serverの公開鍵)

*1 文献20)においても筆者らは秘匿回路計算を用いた分析実験の報告を行っているが、本論文で報告する分析実験の方が先行して実施された。なお実験の内容は大きく異なる。

Proxy の入力: $m_0, m_1 \in G$

Server の入力: SK_S (PK_S に対応する秘密鍵)

Server の出力: m_σ

- (1) Client は以下を行う.
 - (a) $u \in_R \mathbb{Z}/q\mathbb{Z}$ を生成し, Server の公開鍵を用いて暗号文 $E_{PK_S}(\phi(u))$ を計算する.
 - (b) $PK_\sigma = g^u, PK_{1-\sigma} = z/PK_\sigma$ を計算する.
 - (c) PK_0 および $E_{PK_S}(\phi(u))$ をそれぞれ Proxy および Server に送信する.
- (2) Proxy は $PK_1 = z/PK_0$ および $C = (E_{PK_0}(C(m_0)), E_{PK_1}(C(m_1)))$ を計算し, C を Server に送信する.
- (3) Server は $E_{PK_S}(\phi(u))$ を復号して u を取得し, u を秘密鍵としてエラー訂正 C^{-1} により $E_{PK_0}(C(m_0))$ および $E_{PK_1}(C(m_1))$ のいずれかを復号する. これにより, m_σ は正しく復元され, $m_{1-\sigma}$ はエラーとなり復元できない*1.

2.2 歪曲回路

Yao の 2 パーティ秘匿回路計算は, 目的の演算を実行する論理回路を一方のパーティ A が歪曲し, もう一方のパーティ B が当該歪曲回路 (Garbled Circuit) を用いて目的の演算を行う暗号プロトコルである. 文献 16) で提案されている委託型 2 パーティ秘匿回路計算の要素技術であり, 図 1 ではパーティ A が Proxy, パーティ B が Server となる. 以下で示す例は, パーティ A が論理回路 f への入力 $a \in \{0, 1\}^*$ を所持し, パーティ B は a を知ることなく $f(a)$ を計算する. 論理回路 f は, 2 ビット入力 1 ビット出力のゲート $g: \{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\}$ によって構成され, 各ゲートはワイヤ w で結合される. たとえば $a = (a_3, a_2, a_1, a_0) \in \{0, 1\}^4$, $f(a) = a_3 \wedge a_2 \wedge a_1 \wedge a_0$ としたとき, 論理回路 f は図 2 のように構成できる. 図 2 では, たとえば, $(w_0, w_1), w_{01}$ はそれぞれ g_{01} の入力ワイヤ, 出力ワイヤとなり, g_{01} の入出力はそれぞれ $(a_0, a_1), g_{01}(a_0, a_1)$ となる. また w_{01} は g_{01} の出力ワイヤであると同時に g_{0123} の入力ワイヤでもある.

以下に歪曲回路の生成および実行の例を示す.

- (1) パーティ B は以下の処理を行い歪曲回路を生成する.
 - (a) 論理回路 f のすべてのワイヤ w_i に対し, $W_i^0, W_i^1 \in_R \{0, 1\}^\kappa$ を生成し,

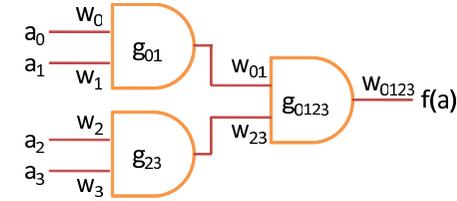


図 2 論理回路の例
Fig. 2 An example of logic circuits.

$(W_i^0, c_i), (W_i^1, \bar{c}_i)$ をそれぞれワイヤ w_i を流れるビット 0, 1 に対応する歪曲値 (Garbled Value) として割り当てる. ここで κ はセキュリティパラメータ, また c_i はランダムビットであり $\bar{c}_i = 1 - c_i$ を満たす.

- (b) 入出力ワイヤ $(w_i, w_j), w_k$ を持つゲート g に対し, 以下の 4 値をランダムな順序に並べたエントリを持つ歪曲真理値表 T_g を割り当てる.

$$\begin{cases} c_i, c_j : (W_k^{g(0,0)}, g(0,0) \oplus c_k) \oplus F_{W_i^0}(c_j) \oplus F_{W_j^0}(c_i) \\ c_i, \bar{c}_j : (W_k^{g(0,1)}, g(0,1) \oplus c_k) \oplus F_{W_i^0}(\bar{c}_j) \oplus F_{W_j^1}(c_i) \\ \bar{c}_i, c_j : (W_k^{g(1,0)}, g(1,0) \oplus c_k) \oplus F_{W_i^1}(c_j) \oplus F_{W_j^0}(\bar{c}_i) \\ \bar{c}_i, \bar{c}_j : (W_k^{g(1,1)}, g(1,1) \oplus c_k) \oplus F_{W_i^1}(\bar{c}_j) \oplus F_{W_j^1}(\bar{c}_i) \end{cases} \quad (1)$$

T_g の各エントリのコロンで区切られた 2 データについて, 左側の 2 ビットデータ $((c_i, c_j)$ 等) をラベルと呼び, 右側の第 2 項および第 3 項の排他的論理和 $(F_{W_i^0}(c_j) \oplus F_{W_j^0}(c_i)$ 等) をマスクと呼ぶことにする. $F_W: \{0, 1\}^{\kappa+1} \rightarrow \{0, 1\}^{\kappa+1}$ は擬似ランダム関数とする. w_k が論理回路の出力を返すワイヤの場合は $c_k = 0$ とする.

- (c) T_g で構成された歪曲回路をパーティ A に送信する.
- (2) パーティ A は歪曲回路を実行するために, 入力 a の各ビット b を入力するワイヤ w_i について, パーティ B と 1-out-of-2 紛失通信を実行し, パーティ B に b を知られることなく $(W_i^b, b \oplus c_i)$ を得る.
- (3) パーティ A は入出力ワイヤ $(w_i, w_j), w_k$ を持つゲート g に割り当てられた歪曲真理値表 T_g について繰り返し以下を行い, 最終的に計算結果 $f(a)$ を得る.
 - (a) w_i, w_j に対応する歪曲値 $(W_i^b, b \oplus c_i), (W_j^{b'}, b' \oplus c_j)$ を入力する.
 - (b) T_g のうち, $(b \oplus c_i, b' \oplus c_j)$ と一致するラベルのエントリを取り出し, 当該

*1 (g, z) から $z = g^v$ を満たす v を求めること (すなわち離散対数問題) が困難であれば, Server が本手続きによって $C(m_0), C(m_1)$ の両方を得ることは困難であることが証明されている.

エントリのマスクを入力の変曲値から生成して打ち消すことで、 w_k の変曲値 $(W_k^{g(b,b')}, g(b,b') \oplus c_k)$ を求め出力する。

- (c) 論理回路の出力を返すワイヤの変曲値の最終ビット(変曲値が $(W_k^{g(b,b')}, g(b,b') \oplus c_k)$ であれば $g(b,b') \oplus c_k$) をあらかじめ定められた順序で結合したものを $f(a)$ とする。

歪曲回路は、回路の中間出力が歪曲値となっているため、入力データだけでなくゲートのロジックの秘匿にも利用することができる。ただしワイヤの配置は既知となるため、ゲート単位ではロジックを秘匿できても回路としてどの程度秘匿できているかは注意が必要である。

Naor らによる委託型 2 パーティ秘匿回路計算は、上記の変曲回路の生成および実行手続きにおいて、パーティ A およびパーティ B をそれぞれ Server, Proxy とし、1-out-of-2 紛失通信の手続きを 2.1 節で説明した中継型 1-out-of-2 紛失通信に置き換えたものである。

3. 提案方式

紛失通信プロトコルを不要とした委託型 2 パーティ秘匿回路計算方式を提案する。紛失通信プロトコルは 2.1 節で説明したように一般に公開鍵暗号または類似の処理を必要とし、2.2 節で述べた歪曲回路の実行の大部分の計算を占める可能性があることから、提案方式の効果は非常に大きいと考えられる。実際、文献 19) における実装結果によれば、Semi-Honest モデルにおける 32 ビット整数の比較演算 (Table 1) は紛失通信プロトコルの処理時間が支配的となっている。また Semi-Honest モデルにおける AES 回路演算 (Table 2) では、紛失通信プロトコルの処理時間が 25~29%程度を占めている。また、たとえば文献 19) の Table 2 Semi-Honest モデルにおける ROM-GRR 方式では、紛失通信プロトコルは 1 回あたりおよそ 20.8 ミリ秒を要し、一方、紛失通信プロトコルを除いた歪曲回路実行処理は 1 ゲートあたりおよそ 0.116 ミリ秒を要すると見積もることができる。したがって単純に考えれば、紛失通信プロトコルを不要とできる提案方式は、25%以上の処理時間の短縮が期待できる。なお提案方式における Client の計算コストは、Naor らの方式よりも優れ、通信コストは同等である。ただし安全性に関しては Naor らの方式よりも強い仮定が要求される。詳しくは 4 章で論じる。また付録に、Free XOR³⁹⁾ と呼ばれる、歪曲回路の生成および実行の計算コストや、歪曲回路の通信コストを削減できる方式について考察する。

提案方式は、各主体が正しく処理を行う Semi-Honest モデルを仮定している。文献 16) では Cut-and-Choose 法を用いて歪曲回路の正当性を検証する方式が述べられており、最

近では不正処理に耐性を持たず様々な方式が提案されているが^{40)–42)}、本論文では以下の 2 つの理由により、そのような方式の検討については議論しないものとする。

- 7 章で考察しているアプリケーションモデルでは、運用対処等により計算主体の不正処理の影響を抑えることができ、オーバーヘッドの大きい不正対策技術を必ずしも必要としない。
- Cut-and-Choose 法等、既存の技術をそのまま適用できるものもあり、またアプリケーションモデルを重視した本論文の主旨からも、処理の説明が煩雑になる不正対策技術の記述はできるだけ最小限にとどめたい。

3.1 プロトコル

提案方式の特徴は、Client からのデータを入力するワイヤの変曲値を Proxy ではなく Client 自身が生成することであり、これにより容易に紛失通信を不要とできる。具体的には以下のプロトコルを実行する。

Client i の入力: $a_i \in \{0, 1\}^*$, PK_S (Server の公開鍵^{*1})

Server の入力: SK_S (PK_S に対応する秘密鍵)

Server の出力: $f(a_1, \dots, a_N)$

- (1) Client i は入力 a_i の各ビット b について、歪曲値 (W^0, c) , (W^1, \bar{c}) を生成し、 $(W^b, b \oplus c)$ を PK_S で暗号化し、 (W^0, c) , (W^1, \bar{c}) および $(W^b, b \oplus c)$ の暗号文 C をセキュア通信路より Proxy に送信する。
- (2) Proxy は以下の処理を行い歪曲回路を生成する。
 - (a) ビット b を入力するワイヤ w に対し、Client より得た (W^0, c) , (W^1, \bar{c}) をそれぞれワイヤ w を流れるビット 0, 1 に対応する歪曲値として割り当てる。ビット b を入力するワイヤ w が複数ある場合は、2 つ目以降について以下の 2 値をランダムな順序に並べたエントリを持つ変換表を作成し、歪曲値を変換する。

$$\begin{cases} c: (W^0, c') \oplus F_{W^0}(c) \\ \bar{c}: (W^1, \bar{c}') \oplus F_{W^1}(\bar{c}) \end{cases} \quad (2)$$

ここで $W^0, W^1 \in \{0, 1\}^\kappa$, $c' \in_R \{0, 1\}$ とする。

- (b) 論理回路 f のワイヤのうち、ビット b を入力としないすべてのワイヤ w_i に対し、 $W_i^0, W_i^1 \in_R \{0, 1\}^\kappa$ を生成し、 (W_i^0, c_i) , (W_i^1, \bar{c}_i) をそれぞれワイヤ w_i

*1 任意の公開鍵暗号アルゴリズムでよい。

を流れるビット 0, 1 に対応する歪曲値として割り当てる．

- (c) 入出力ワイヤ (w_i, w_j) , w_k を持つゲート g に対し, 式 (1) をランダムな順序に並べたエントリを持つ歪曲真理値表 T_g を割り当てる．
 - (d) T_g で構成された歪曲回路および各 Client から受信した暗号文 C を Server に送信する．
- (3) Server は C を復号して回路への入力ビット b に対応する歪曲値 $(W^b, b \oplus c)$ を取得し, ビット b を入力するワイヤ w が複数ある場合は, 2 つ目以降について式 (2) に基づき歪曲値を変換する．そして入出力ワイヤ (w_i, w_j) , w_k を持つゲート g に割り当てられた歪曲真理値表 T_g について繰り返し以下を行い, 最終的に計算結果 $f(a_1, \dots, a_N)$ を得る．
- (a) w_i, w_j に対応する歪曲値 $(W_i^b, b \oplus c_i)$, $(W_j^b, b \oplus c_j)$ を入力する．
 - (b) T_g のうち, $(b \oplus c_i, b' \oplus c_j)$ と一致するラベルのエントリを取り出し, 当該エントリのマスクを入力歪曲値から生成して打ち消すことで, w_k の歪曲値 $(W_k^{g(b, b')}, g(b, b') \oplus c_k)$ を求め出力する．
 - (c) 論理回路の出力を返すワイヤの歪曲値の最終ビット(歪曲値が $(W_k^{g(b, b')}, g(b, b') \oplus c_k)$ であれば $g(b, b') \oplus c_k$) をあらかじめ定められた順序で結合したものを $f(a_1, \dots, a_N)$ とする．

上記の手続きは, ビット b を入力するワイヤ w が複数ある場合にやや複雑となるが, これは Client の負担を Proxy および Server に分散して軽減させるためのものであり, 式 (2) の (W^0, c') , (W^1, \bar{c}') は Client が生成して Proxy に送信し, 歪曲値 $(W^b, b \oplus c')$ は Server が得るような手続きとしてもよい．

4. 安全性評価

提案方式における Client i の情報 a_i の秘匿性について, Naor らの方式と対比しながら考察する．

提案方式は Proxy および Server に対して Semi-Honest モデル, すなわち Proxy および Server は正しくプロトコルを実行するものと仮定する．また Proxy および Server の結託はないものとする．ただし Client による不正データの送信や, Proxy または Server が一部の Client と結託して別の Client の情報の秘匿性を侵害しようとする攻撃については考慮する．以下では, Server が他の主体と結託しない限り Client の情報を秘匿できることを示す．Naor らの方式は Server (Proxy も同様) が一部の Client と結託しても別の Client の情報

を秘匿できるため, 提案方式はより強い安全性の仮定が要求される．

まず, すべての Client が正しいデータを送信した場合について考察する．Proxy が受信するデータは, Client からの送信データ (W^0, c) , (W^1, \bar{c}) および暗号文 C である． (W^0, c) , (W^1, \bar{c}) は明らかに a_i と独立であり, C は任意の公開鍵暗号によるものとしているため, 意味的安全 (Semantically Secure) な方式であれば, Proxy が受信データから a_i を推定することは明らかに困難である．

Server が受信するデータは, Client から Proxy を経由して送信される暗号文 C , および Proxy が生成する歪曲回路である．このとき C を復号して得られる歪曲値 $(W^b, b \oplus c)$ は, Naor らの方式において, Server が Proxy および Client と中継型 1-out-of-2 紛失通信を実行して得られる歪曲値に等しい．歪曲回路は, ビット b を入力するワイヤ w が複数ある場合を除き, Naor らの方式同様, Yao の歪曲回路を単純に用いる．ビット b を入力するワイヤ w が複数ある場合であっても, 式 (2) の (W^0, c') , (W^1, \bar{c}') を Client が生成して Proxy に送信したものとみれば, Yao の歪曲回路に等しい．

次に Client による不正データ送信の対策について考察する．Client は多数からなる場合があり, Client に対して Semi-Honest モデルを仮定することは現実的でない場合がありうる．ここで不正データとは, Proxy に送信するデータ (W^0, c) , (W^1, \bar{c}) および Server が受信する暗号文 C について, C の復号結果が (W^0, c) , (W^1, \bar{c}) のいずれでもない場合である．これにより計算の妨害が容易に可能となる．この種の攻撃を防ぐためには, Client は (W^0, c) および (W^1, \bar{c}) のハッシュ値を適当に順序を入れ替えて公開すればよい．そして Proxy は受信した (W^0, c) , (W^1, \bar{c}) のハッシュ値がともに公開されていることを確認し, Server は C の復号結果のハッシュ値が公開されていることを確認する．

最後に, Proxy または Server が一部の Client と結託して別の Client の情報の秘匿性を侵害しようとする攻撃について考察する．Client が Proxy に送信するデータ (W^0, c) , (W^1, \bar{c}) は Client ごとに独立の値である．したがって Proxy と一部の Client の結託により別の Client の情報の秘匿性を侵害することはできない．Server が一部の Client と結託した場合, Server は Proxy から受信した歪曲回路の一部を結託した Client の情報に基づき復元できる．ここで Server は Client i と結託し, Client j の情報を得ようとする, すなわち Server は 3.1 節のプロトコルの手続き (1) の (W_i^0, c_i) , (W_i^1, \bar{c}_i) , $(W_j^b, b \oplus c_j)$ から b を求める場合を考える．すると式 (1) の 4 式のうち, 1 番目と 3 番目 ($b = 0$ の場合), または 2 番目と 4 番目 ($b = 1$ の場合) のマスクを計算でき, 当該マスクを打ち消すことで $(W_k^{g(0, b)}, g(0, b) \oplus c_k)$, $(W_k^{g(1, b)}, g(1, b) \oplus c_k)$ を得ることができる．ここでたとえば $g(x, y) = x \vee y$, すなわち g

を論理和 (OR) ゲートとしたとき, $b = 0$ であれば $W_k^{g(0,b)} \neq W_k^{g(1,b)}$, $b = 1$ であれば $W_k^{g(0,b)} = W_k^{g(1,b)}$ となるため, $W_k^{g(0,b)}$, $W_k^{g(1,b)}$ の等号判定を行うことで b を識別できることが分かる. 対策として, 2.2 節で述べたようにゲートのロジックを秘匿する方法が有効と考えられるが, 効果の程度や実現可能性は現時点では明らかでなく, またロジックを秘匿することで新たな攻撃が発生する可能性もあり, 今後の検討課題としたい.

以上から, 提案方式では Server が他の主体と結託しない限り Client の情報を秘匿できることを確認した. ただし前述のとおり, Naor らの方式は Server (Proxy も同様) が一部の Client と結託しても別の Client の情報を秘匿できるため, 提案方式はより強い安全性の仮定が要求される. そこで 7 章では, アプリケーションの観点から本仮定の影響について論じる.

5. 実装

本章では, 3.1 節で述べた提案方式に基づく実装システムについて述べる. 図 3 にシステム構成を示す. 図 1 における Client での処理は, Microsoft Excel^{*1} のプラグインとして実装しており, 利用者が Excel 上において秘匿したい情報を矩形選択し, メニュー上から「秘匿処理実行」を選んで実行する. その結果, 当該情報を秘匿したデータが新しいシートに出力される. 複数の Client において秘匿したデータは, 特定の Client (以降, 代表 Client と呼ぶこととする) が受信して Excel 上で統合し, 秘匿回路計算の対象となるデータを矩形選択したうえで, Excel のメニュー上から「秘密計算実行」を選んで実行する. この際に, 代表 Client はサブメニュー上から「演算種別」を選択する. 現在の実装において実行が可能な演算種別は以下の 11 種類である.

- 最大値/最小値/中央値/クロス集計/平均値/最頻値/分散/加算/減算/乗算/平方算

ここで加減乗算および平方算は他の統計演算の要素演算として用いている. 最大値と最小値は本質的に同様の処理を行う. なお中央値, 最頻値, および分散は基礎としている回路のアルゴリズムの効率が悪いので, 今後の検討課題とし, 今回計測は行っていない.

Proxy および Server は Linux 上で動作するサーバプログラムとなっており, 3.1 節で述べたとおり, それぞれ歪曲回路の生成および実行を行い, 計算結果を代表 Client へ返却する. 代表 Client では, Excel 上に新たなシートを作成し, 計算結果を表示する.

*1 Microsoft, Excel はともに米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です.

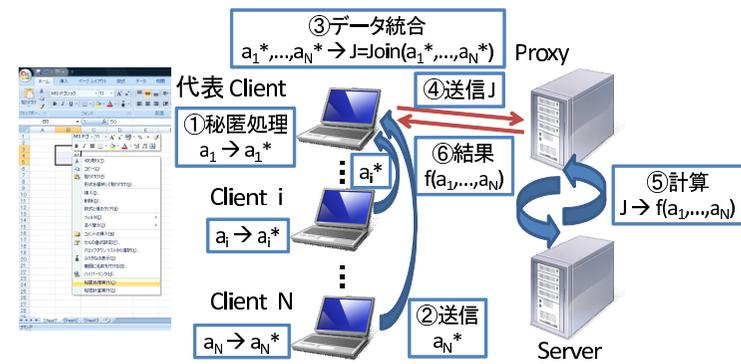


図 3 秘匿回路計算実装システム

Fig. 3 Implementation system of our secure circuit evaluation.

なお, Proxy が利用する Client からの受信データは, 代表 Client が中継する処理フローとなっているため, 当該受信データは Proxy の公開鍵を用いて暗号化する設計としている. ここで暗号化するデータは元のデータのおよそ $3(\kappa + 1)$ 倍となるが, 元のデータの暗号化は共通鍵暗号 (Camellia-128 ビット) を用い, 共通鍵を公開鍵暗号で暗号化する仕様としているため, 暗号化にかかるオーバーヘッドはそれほど大きくない (Excel 上で 20 ビット, 10,000 データの場合で乱数生成を含め 15 秒程度).

5.1 性能評価

本節では実装した 2 パーティ秘匿回路計算システムを用いて性能評価を行った結果を示す. 実験環境は表 1 に示すとおりである. 本評価では最大値計算, クロス集計, および平均値計算について, 入力データの数とビット数をパラメータとし, 全体の処理時間を計測した. 評価結果を表 2, 表 3, 表 4, 表 5, 表 6 に示す. 特に最大値計算およびクロス集計はテストベッドとして事前処理の有無のほか, 参考までにゲート数, 回路の深さ等, 詳細情報を記述した. クロス集計および平均値計算は事前計算なしの処理時間のみを記載した. 表中の計算時間 (1) は秘匿回路計算に必要な処理をすべて行った場合に要する時間, 計算時間 (2) は論理回路 f が与えられているという前提で乱数生成, 秘匿回路生成を事前に行っていた場合に要する時間である. それぞれ 3 回の計測を行い, 平均値を実験結果としている.

計算結果より, 平均値や最大値計算は入力データならびに入力ビット数にほぼ比例して計算時間が増大していることが分かる. ただしクロス集計は入力ビット数に対して指数的に計算時間が増大するアルゴリズムを用いて実装しており, 効率が悪い. 実際, 10,000 入

表 1 性能評価環境

Table 1 Implementation environment.

CPU	Intel Core2 Quad Q9650 (3.0 GHz)
メモリ	4 GB
NIC	1000BASE-T
サーバ OS	Fedora 10 (Linux Kernel 2.6)
暗号ライブラリ	Camellia-128-ecb in the OpenSSL Library Ver. 0.9.8 i

表 2 性能評価結果 (最大値): 5 ビット入力

Table 2 Result of processing time for computing maximum values (5 bits input).

入力データ数	100	1,000	10,000
ゲート数	3,970	39,970	399,970
回路の深さ	58	82	114
計算時間 (1) [msec]	48.47	426.88	4,272.08
計算時間 (2) [msec]	42.35	186.68	1,811.97

表 3 性能評価結果 (最大値): 10 ビット入力

Table 3 Result of processing time for computing maximum values (10 bits input).

入力データ数	100	1,000	10,000
ゲート数	8,435	84,935	849,935
回路の深さ	93	132	184
計算時間 (1) [msec]	99.82	892.96	9,116.12
計算時間 (2) [msec]	87.89	418.05	3,809.19

表 4 性能評価結果 (最大値): 20 ビット入力

Table 4 Result of processing time for computing maximum values (20 bits input).

入力データ数	100	1,000	10,000
ゲート数	17,365	174,865	1,749,865
回路の深さ	163	232	324
計算時間 (1) [msec]	189.91	1,848.11	18,691.41
計算時間 (2) [msec]	126.15	807.63	8,734.44

表 5 性能評価結果 (クロス集計): 5 ビット入力

Table 5 Result of processing time for computing cross tabulations (5 bits input).

入力データ数	100	1,000	10,000
ゲート数	50,964	515,816	5,168,560
回路の深さ	55	106	200
計算時間 (1) [msec]	628.59	6,015.43	60,218.49
計算時間 (2) [msec]	280.05	2,696.16	26,906.37

表 6 性能評価結果 (平均値)

Table 6 Result of processing time for computing average values.

入力データ数	100	1,000	10,000
5 ビット計算時間 (1) [msec]	87.02	357.07	3,509.71
10 ビット計算時間 (1) [msec]	88.61	672.12	6,644.50
20 ビット計算時間 (1) [msec]	151.12	1,285.49	12,706.04

力では 7 ビットの実行時にメモリ不足でスワップが発生し、計測ができなかった。同様に 1,000 入力では 10 ビットでスワップが発生した。なお、乱数生成、秘匿回路生成を事前に行った場合、計算時間を 12.7~58.3%程度削減でき、特に 1,000 入力や 10,000 入力はおおむね 55%以上の削減効果を得る等、十分効果的であることを確認した。

論理回路 1 ゲートあたりの計算時間については、最大値の計算において誤差が最も少ないと思われる 10,000 入力の計測値をゲート数で割れば、乱数生成・秘匿回路生成を事前に行わない場合で平均 $10.7 \mu\text{sec}$ 、事前に行った場合には平均 $4.67 \mu\text{sec}$ の結果を得ることができる。表 2~表 4 から、100 入力の際は事前計算なしとありとの差が小さくなっていることが分かる。これは事前に用意したデータを HDD から読み込むことによって発生するオーバヘッドにより、事前計算の効果が若干低くなっていることによるものと考えられる。文献 49) によれば、80 GB モデルの HDD の遅延は平均 4.2 msec となる。表 2~表 4 から、25~45 msec 程度の遅延が発生していることが分かり、5~10 回程度の読み込みを行っているものと考えられる。なお前述のとおり、データ数が増大するとメモリ不足でスワップが発生する可能性があるため注意が必要である。

Proxy, Server の処理時間の内訳は実装方法の問題で計測できなかった。これについては今後の課題としたいが、以下に簡単な考察を与える。Proxy は生成した乱数を用いて歪曲回路を作成する。乱数はゲートあたり 2 個、歪曲回路はゲートあたり 4 回の疑似ランダム関数演算を行う必要がある。一方 Server は乱数の生成は不要で、ゲートあたり 2 回の疑似ランダム関数演算を行う必要がある。その他の演算処理は乱数生成や疑似ランダム関数演算と比べると小さいものと考えられる。したがって、乱数生成と疑似ランダム関数演算を同等の処理時間とみれば、Proxy は Server よりも事前計算なしの場合 3 倍、ありの場合 2 倍の処理時間を要すると見積もることができる。

6. 実 験

2009 年 10 月 26~28 日に富山国際会議場で行われた、情報処理学会コンピュータセキュ

CSS2009行動分析実験アンケート

所属区分	学生
年齢	19才以下
出身地	北海道
血液型	A型
今熱いと思う研究テーマ (複数回答可)	<input type="checkbox"/> 暗号・評価 <input type="checkbox"/> 署名・暗号プロトコル <input type="checkbox"/> 情報のマイニング <input type="checkbox"/> ネットワーク監視・追跡 <input type="checkbox"/> コンピュータウイルス <input type="checkbox"/> Web・メールセキュリティ <input type="checkbox"/> アクセス制御 <input type="checkbox"/> 認証・バイオメトリクス <input type="checkbox"/> セキュリティ・設計・実装 <input type="checkbox"/> OS・仮想化 <input type="checkbox"/> ハードウェア <input type="checkbox"/> ユビキタスセキュリティ <input type="checkbox"/> 電子盗取/印刷 <input type="checkbox"/> コンテナ保護 <input type="checkbox"/> ソフトウェア保護 <input type="checkbox"/> リスク分析・セキュリティポリシー <input type="checkbox"/> セキュリティ評価・監査 <input type="checkbox"/> 個人情報・プライバシー保護 <input type="checkbox"/> フォレンジクス <input type="checkbox"/> セキュリティ教育・法律 <input type="checkbox"/> SPT(心理学・トラスト) <input type="checkbox"/> マルウェア対策 <input type="checkbox"/> その他
ファイル交換ソフトを利用していますか？	利用していません
好きなお酒の種類	ビール
RFIDカードのPWを入力してください (英字は大文字で入力お願いします)	

図 4 CSS2009 で用いられた秘匿回路計算実行用秘匿アンケート

Fig. 4 Secure questionnaire for running secure circuit evaluation in CSS2009.



図 5 左: RFID 制御ソフトウェア表示画面, 右上: RFID タグ (V750-D22M01-IM), 右下: RFID アンテナ (V750-HS01CA-JP)

Fig. 5 Left: RFID control software display, Upper right: RFID tag (V750-D22M01-IM), Lower right: RFID antenna (V750-HS01CA-JP).

リティ研究会主催「コンピュータセキュリティシンポジウム 2009 (CSS2009)⁴³⁾」で、秘匿回路計算実装システムを用いた行動分析実験を行った。本実験の目的は、(1) シンポジウムの参加者実態調査を把握することで調査結果を研究用途に供する、(2) 今後の運営にフィードバックする、そして (3) 秘匿回路計算を用いたシステムが、実用環境において不具合なく動作することを検証することである。ここでは特に (3) に焦点を当てて考察する。

本実験で分析対象とするために収集した情報は以下の 2 種類である。

- アンケートによる、CSS2009 参加者の属性情報
- RFID システムにより収集される、CSS2009 参加者の行動ログ

アンケートは氏名や住所等、直接本人を特定できる情報は含めず匿名としたが、参加者の属性情報として所属、年齢、出身地等、合計 7 項目をすべて選択式の設問とした。行動ログは、「どの発表セッションの部屋に入退室したか」という情報である。これらの情報を統合すれば、たとえば「暗号の発表セッションを聴講した参加者は 20 代が多い」といった分析結果を得ることが期待される。

6.1 情報収集方法

属性情報は会場の受付等に設置した 4 台の Client PC 端末を介して、参加者にブラウザベースのアンケートに答えてもらうことで収集した (図 4)。

行動ログについては約 3m 程度の距離からタグを検知することのできる、UHF 帯 RFID システムを用いて収集した。RFID はアンテナとタグ間で無線通信を行うことで、アンテナ付近のタグの存在を検知することができる。このアンテナを CSS2009 の 6 つすべてのセッ

ション会場入り口に配置するとともに、実験への協力に同意する参加者に RFID タグを配布し、名札とともに身につけてもらった。これにより各参加者の位置情報、そしてこれを連続的に取得することで行動ログを得ることができる。なお本実験ではタグ検知時刻およびタグ ID を記録しているが、利用者登録は行っていないため、システム管理者であってもタグ ID から即座に参加者と位置情報が紐付くことはない。

本システムでは参加者にカードをかざす等の行為を求めることなくアンテナ前方の通過を検知することができる。参加者が自身の被検知を確認し実感できるように、検知されたタグ ID は会場に設置したモニタにリアルタイム/グラフィカルに表示した (図 5 左)。タグには 128 bits の書き込み可能領域にあらかじめ 1 ~ 400 の ID を書き込み、タグ表面にも当 ID を記載した。なおハードウェアはオムロン社のリーダライタ V750-BB50C04-JP およびアンテナ V750-HS01CA-JP (図 5 右下)、タグ V750-D22M01-IM (図 5 右上) を用いた。ソフトウェアに関しては、オムロンソフトウェア社による評価ライブラリを利用して、C# により制御ソフトウェア (図 5 左) を実装した。

6.2 実験結果

収集したデータ量については表 7 のとおりである。重複を除くタグの検知枚数から、CSS2009 参加者の半数以上が実験への協力をしたこと、またアンケート回答数からは参加者の 1/4 以上がアンケートに回答したことが分かる。分析結果の一例を図 6 に示す。これは行動ログとアンケート結果をクロス集計し、発表セッションと聴講者属性の関係を探ったものである。分析結果の考察については本論文の主旨とは異なるため割愛する。CSS2009

表 7 本実験で収集したデータ
Table 7 Amount of data collected in CSS2009.

CSS2009 参加者	362 名
会期中のタグ検知数 (のべ数)	2,339 回
同上 (重複除く)	215 枚
アンケート回答数	97 名

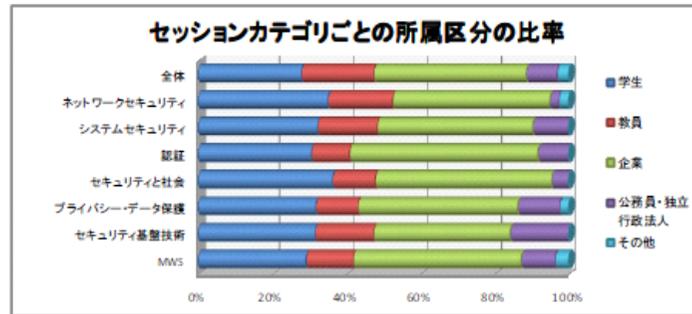


図 6 分析結果の一例
Fig. 6 An example of analysis result.

ホームページ⁴³⁾に記載しているため興味のある方は参照されたい。

本実験では提案方式の実装システムを実際に動作させ分析を行った。この分析では 65 回のクロス集計の秘匿回路計算を行い、Excel 上の操作まで含め 30 分以内に計算が終了した。このことから従来計算コストのために現実的でないと思われていた秘匿回路計算が、上記のような重要な応用例に対してすでに実用に耐えうる性能に達し始めているといえる。

今後はより処理性能を高めることで、ライフログ等のより高度に蓄積されたデータへの適用を目指す予定である。たとえば近年は CPU のマルチコア化、GPU による汎用並列計算環境の出現等により並列計算資源が豊富になっているため、これらの利用が考えられる。また付録 A.1 の技法の適用等も数倍の高性能化を見込むことができる。さらに性能面のみでなく、文献 20) で行ったような、秘匿回路計算の採用によるユーザのデータ提供量意思および収集データ量の変化等の検証、そして今回の実験では未実装のため RFID システムによる行動ログを秘匿して演算することができなかったが、演算の充実によりこのようなデータの秘匿回路計算にも対応する等、適用範囲の拡充等も行っていく予定である。

7. 考 察

7.1 パーソナル情報の安全な活用に向けて

医療分野やサービス分野等では、パーソナル情報の安全活用に向けた各種取組みがなされており、国内ではたとえば総務省、経済産業省、および厚生労働省の連携による“健康情報活用基盤実証事業⁴⁾”や、経済産業省による“情報大航海プロジェクト⁵⁾”において、パーソナル情報の利活用を促進するためのプライバシー保護技術が検討課題としてあげられている。パーソナル情報の扱いは、個人情報保護法の施行にともない、より厳重な管理が求められるようになってきている。したがって事業者からすれば、パーソナル情報をいっさい復元せず利活用できる秘匿関数計算は、管理コストの面で優れるものと考えられる。またパーソナル情報を提供する個人の立場としても、秘匿関数計算の効果は、事業者からの漏洩リスク低下に加え、事業者が許諾なくパーソナル情報を利用することを防止できる点でも大きい。

以下では、提案方式のモデルである委託型 2 パーティ秘匿回路計算がどのようにパーソナル情報の安全活用に効果的に適用できるかについて考察する。

パーソナル情報の活用を望む主体 (分析主体) は、図 3 における代表 Client として、各 Client から秘匿されたパーソナル情報を収集する。この時点で、分析主体からの機密情報漏洩のリスクは最小限に抑えることができる。なお Client は一般消費者でもよく、顧客情報を保管している事業者であってもよい。また分析主体は Server も兼ね、Proxy は第三者機関が運営する。そしてパーソナル情報の活用時は、分析主体は第三者機関と 3.1 節で示した手続きにより歪曲回路の生成および実行を行い、代表 Client として結果を得る。なお分析主体の不正を防ぐため、第三者機関に送信される Client からの暗号文には、復号するとパーソナル情報の利用範囲を示した情報が付加されるようにする。これにより、分析主体による許可されていないパーソナル情報の活用を防ぐことができる。一方、第三者機関は Proxy の役割のみ担っているため、3.1 節で示した手続きによって入力したパーソナル情報を推定することは困難である。すなわち第三者機関からの情報漏洩リスクも最小化される。第三者機関は分析主体に不正データを送信することで計算結果を改ざんすることは可能となるが、それによる実用上のメリットはほとんどないものと考えられる。

4 章で述べたように、提案方式は Server (上記の例では分析主体) が一部の Client (上記の例では一般消費者または事業者) と結託した場合、別の Client のパーソナル情報が漏れる可能性がある。しかし分析主体の漏洩リスクを考えると、分析主体はパーソナル情報を所持することなく所望の分析結果を得ることができるため通常と比べ漏洩リスクは下がり、

また第三者機関を含む別の主体に預託する情報はパーソナル情報を秘匿したデータであるため、別の主体から漏洩するリスクも下がる。したがって分析主体にとっても提案方式の適用効果は十分あるものと考えられる。

処理性能については、分析の基本的演算である集計をベースに考察する。集計結果は統計量であり、それだけではプライバシーを侵害するリスクは少ないため、集計結果を秘匿することなく扱い、高度な統計分析やデータマイニング等に活用することが可能となる。表5では、5ビットデータのクロス集計結果について示しているが、これは1ビットデータの集計(カウンタ)を32回繰り返し実行した結果にほぼ等しい。すなわち、たとえば10,000入力データであれば、表5からカウンタの処理1回あたり $60.218/32 \approx 1.9$ 秒を要する計算となる。また表5の結果からも分かるように、処理時間はおよそ入力データ数に比例することから、1,000,000入力データのカウンタの処理でも3分強と、現実的な時間で処理可能である。ただし現時点の実装システムでは、メモリの制約条件やExcelの仕様の問題から10,000入力であれば6ビット以内に制限される。6ビットは単純なクロス集計表であれば表現可能なレベルであると考えられる。しかしより複雑な表を得るためには技術改善が必要である。

一方 Client の計算コストについては、Naor らの方式では 2.1 節から分かるようにパーソナル情報のビット長の回数だけ公開鍵の生成および公開鍵暗号化処理が必要となるのに対し、提案方式では、Server が復号して得る Client のデータ(3.1 節の手続き(1)に記載の、パーソナル情報のビット長の個数の歪曲値)は共通鍵暗号により一括して暗号化し、単一の共通鍵のみ Server の公開鍵で暗号化してもよい。したがって、提案方式は Naor らの方式と比べ Client の計算コストの大幅な削減が期待でき、これは計算資源の乏しいセンサデバイスを用いてパーソナル情報を収集する場合等において特に有効となると考えられる。

以上、パーソナル情報の安全活用について、3.1 節で示した提案方式を適用した実用性の高いアプリケーションモデルを述べた。本節で述べたモデルを図7に示す。

7.2 クラウドコンピューティングの機密情報保護の実現に向けて

先に述べたとおり、グローバルに拡散した計算資源を用いてサービス向上を図るクラウドコンピューティングが管理コストの低下や利便性向上の面で注目を集めているが、一般に外部に情報を預けることから、情報保護に対する問題解決は重要なセキュリティ課題であるといえる。特にクラウドコンピューティングにおいては、拡散した計算資源を用いることから運用による徹底したセキュリティ対策を実現することは容易ではなく、また、組織がネットワークを通じて機密情報を外部に預ける行為自体が組織のセキュリティポリシーに反することも十分に考えられ、場合によってはクラウドコンピューティングの普及阻害要因となる可能

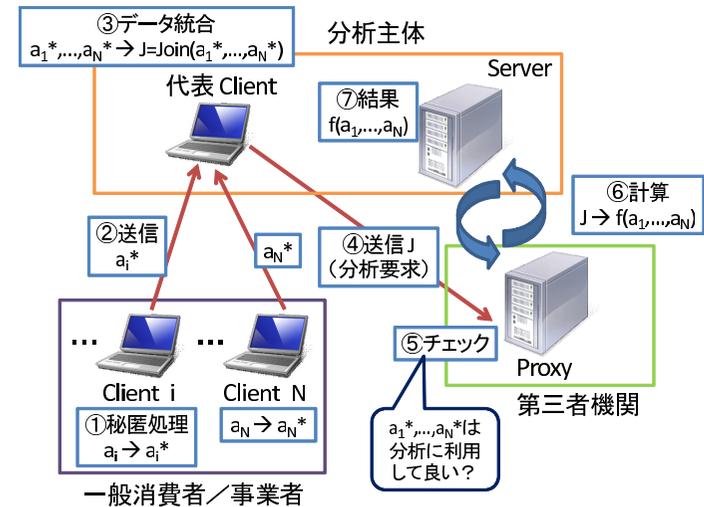


図7 提案方式を用いてパーソナル情報の安全活用を実現するためのシステム構成

Fig. 7 System configuration to achieve safety use of personal information by using proposal method.

性もある。このような背景から、秘匿関数計算はパーソナル情報の安全活用と同様に、クラウドコンピューティングにおける情報保護技術としても有効であると考えられる。

以下では、前節同様、提案方式のモデルである委託型2パーティ秘匿回路計算がどのようにクラウドコンピューティングにおける情報保護技術として効果的に適用できるかについて考察する。

クラウドコンピューティングにおいては、2パーティ委託型秘匿回路計算の応用モデルとして、図8のように「半委託」型によって自組織が保有する機密情報の安全な保管と活用の両立が期待できる。すなわち、組織は Client と Server を兼ね、Proxy は第三者機関が運営し、機密情報を秘匿して Proxy に送信した後に機密情報を削除し、活用時は第三者機関との協調計算により必要な情報を得る。この第三者機関の機能をクラウドサーバとして実装することで、安全かつ利便性の高いクラウドコンピューティングの実現が期待できる。なお本モデルにおいては Client は単一であり、かつ Client と Server は同一組織としていることから、Client の情報の秘匿性について提案方式が仮定している「Server は他の主体と結託しない」ことは十分妥当な仮定であるといえる。

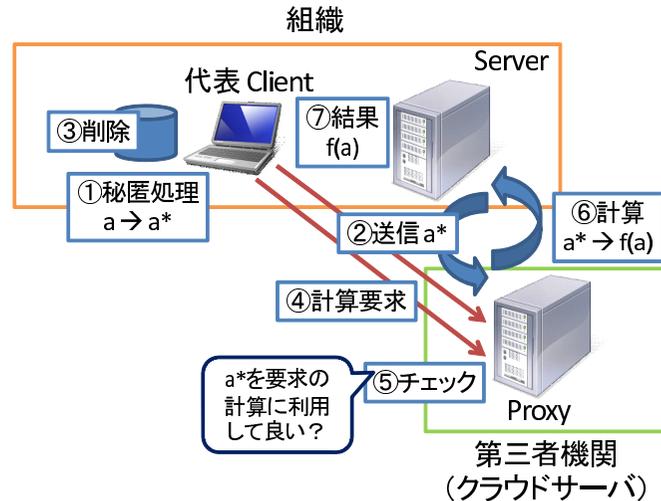


図 8 クラウドコンピューティングに適した 2 パーティ委託型秘匿回路計算の応用モデル

Fig. 8 An application model of delegation-based 2-party secure circuit evaluation applicable to cloud computing.

8. まとめと今後の課題

パーソナル情報の安全な活用やクラウドコンピューティングにおける機密情報保護の実現に向け、本論文では入力データや演算ロジックを秘匿しつつ各種情報処理を可能とする秘匿回路計算技術に着目し、委託型 2 パーティモデルの効率的な方式を提案した。さらに、実装や実験により提案方式の有用性を定量的に示すとともに、委託型 2 パーティモデルがパーソナル情報の安全な活用やクラウドコンピューティングにおける機密情報保護に有効であることを明らかにした。現状の実装システムは統計分析への適用を主眼としており基本的な統計演算のみ実装しているが、今後はクラウドコンピューティングをより意識した情報処理への適用について検討していく予定である。

謝辞 提案方式の安全性評価の誤りや、先行技術ならびに実験結果に関する説明や考察の不足、その他多くの記述の不備について、ご指摘およびコメントいただきました査読者の方々に深く感謝いたします。

参考文献

- 1) Chida, K. and Takahashi, K.: Privacy Preserving Computations without Public Key Cryptographic Operation, *Proc. IWSEC 2008*, LNCS 5312, pp.184–200, Springer-Verlag (2008).
- 2) 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, コンピュータセキュリティシンポジウム 2009 (CSS2009) 論文集, pp.625–630, 情報処理学会 (2009).
- 3) 五十嵐大, 千田浩司, 柴田賢介, 山本太郎, 高橋克巳: 2 パーティ秘匿回路計算を利用したプライバシー保護データ分析実験報告 (1) — CSS2009 における行動分析, 情報処理学会研究報告, Vol.2010-CSEC-48, No.2, pp.1–8 (2010).
- 4) アクセンチュア: 健康情報活用基盤構築のための標準化及び実証事業, 入手先 (<https://microsite.accenture.com/meti/Pages/default.aspx>) (参照 2011-03-17).
- 5) 経済産業省: 情報大航海プロジェクト, 入手先 (http://www.meti.go.jp/policy/it_policy/daikoukai/igvp/index/index.html) (参照 2011-03-17).
- 6) 経済産業省: 「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, 入手先 (<http://www.meti.go.jp/press/20080703007/20080703007.html>) (参照 2011-03-17).
- 7) GeoPKDD: Geographic Privacy-aware Knowledge Discovery and Delivery, available from (<http://www.geopkdd.eu/>) (accessed 2011-03-17).
- 8) Electronic Health Information Laboratory: KnowledgeBase, available from (<http://www.ehealthinformation.ca/knowledgebase/>) (accessed 2011-03-17).
- 9) ENISA: Cloud Computing Information Assurance Framework, available from (<http://www.enisa.europa.eu/act/rm/files/deliverables/cloud-computing-information-assurance-framework>) (accessed 2011-03-17).
- 10) CSA: Security Guidance for Critical Areas of Focus in Cloud Computing V2.1, available from (<https://cloudsecurityalliance.org/csaguide.pdf>) (accessed 2011-03-17).
- 11) ASP・SaaS の情報セキュリティ対策に関する研究会: ASP・SaaS における情報セキュリティ対策ガイドライン, 入手先 (http://www.soumu.go.jp/menu_news/s-news/2008/pdf/080130_3_bt3.pdf) (参照 2011-03-17).
- 12) Yao, A.C.: Protocols for Secure Computations, *Proc. FOCS '82*, pp.160–164, IEEE (1982).
- 13) Lindell, Y. and Pinkas, B.: Privacy Preserving Data Mining, *Proc. CRYPTO 2000*, LNCS 1880, pp.36–54, Springer-Verlag (2000).
- 14) Aggarwal, C.C. and Yu, P.S.: *Privacy-Preserving Data Mining: Models and Algorithms*, Springer-Verlag (2009).
- 15) Vaidya J., Clifton, C.W. and Zhu, Y.M.: *Privacy Preserving Data Mining*,

- Springer-Verlag (2005).
- 16) Naor, M., Pinkas, B. and Sumner, R.: Privacy Preserving Auctions and Mechanism Design, *Proc. ACM EC '99*, pp.129–139, ACM (1999).
 - 17) Even, S., Goldreich, O. and Lempel, A.: A Randomized Protocol for Signing Contracts, *Comm. ACM*, Vol.28, No.6, pp.637–647, ACM (1985).
 - 18) Malkhi, D., Nisan, N., Pinkas, B. and Sella, Y.: Fairplay – A Secure Two-Party Computation System, *Proc. USENIX Security 2010*, pp.287–302, USENIX (2004).
 - 19) Pinkas, B., Schneider, T., Smart, N.P. and Williams, S.C.: Secure Two-Party Computation Is Practical, *Proc. ASIACRYPT 2009*, LNCS 5912, pp.250–267, Springer-Verlag (2009).
 - 20) 柴田賢介, 千田浩司, 山本太郎, 高橋克巳, 金井 敦: 2 パーティ秘匿回路計算を利用したプライバシー保護データ分析実験報告(2) —大学生の成績と生活実態との相関分析, 情報処理学会研究報告, Vol.2010-CSEC-48, No.3, pp.1–6 (2010).
 - 21) Ben-David, A., Nisan, N. and Pinkas, B.: FairplayMP: A System for Secure Multi-Party Computation, *Proc. ACM CCS 2008*, pp.257–266, ACM (2008).
 - 22) Bogetoft, P., Christensen, D.L., Damgård, I., Geisler, M., Jakobsen, T., Kroigaard, M., Nielsen, J.D., Nielsen, J.B., Nielsen, K., Pagter, J., Schwartzbach, M. and Toft, T.: Secure Multiparty Computation Goes Live, *Proc. FC 2009*, LNCS 5628, pp.325–343, Springer-Verlag (2009).
 - 23) 独立行政法人情報通信研究機構(NICT): 報道発表(お知らせ)スピア型サイバー攻撃判定システム開発のための共同実証実験を開始—特定の組織に限定したサイバー攻撃を早期に検知するシステムの実現に向けて, 入手先 (http://www2.nict.go.jp/pub/whatsnew/press/h19/080303/080303_1.html) (参照 2011-03-17).
 - 24) 佐藤夏樹, 篠原靖志: プライバシーを保護した需要データ収集・共用方式の開発(その1) — 需要データ収集・需要特性算出方式, 電力中央研究所システム技術研究所研究報告, 報告書番号: R07028 (2008).
 - 25) Yao, A.C.: How to generate and exchange secrets, *Proc. FOCS '86*, pp.162–167, IEEE (1986).
 - 26) Goldreich, O., Micali, S. and Wigderson, A.: How To Play Any Mental Game or A Completeness Theorem for Protocols with Honest Majority, *Proc. STOC '87*, pp.218–229, ACM (1987).
 - 27) Chaum, D.L.: Untraceable Electronic Mail, Return Address, and Digital Pseudonyms, *Comm. ACM*, Vol.24, No.2, pp.84–88, ACM (1981).
 - 28) Jakobsson, M. and Juels, A.: Mix and Match: Secure Function Evaluation via Ciphertexts, *Proc. ASIACRYPT 2000*, LNCS 1976, pp.162–177, Springer-Verlag (2000).
 - 29) Gentry, C.: Fully Homomorphic Encryption Using Ideal Lattices, *Proc. STOC '09*, pp.169–178, ACM (2009).
 - 30) Ben-Or, M., Goldwasser, S. and Wigderson, A.: Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation, *Proc. STOC '88*, pp.1–10, ACM (1988).
 - 31) Chaum, D., Crepeau, C. and Damgård, I.: Multiparty Unconditionally Secure Protocols, *Proc. STOC '88*, pp.11–19, ACM (1988).
 - 32) Cramer, R., Damgård, I. and Nielsen, J.B.: Multiparty Computation from Threshold Homomorphic Encryption, *Proc. EUROCRYPT 2001*, LNCS 2045, pp.280–300, Springer-Verlag (2001).
 - 33) Schoenmakers, B. and Tuyls, P.: Practical Two-Party Computation based on the Conditional Gate, *Proc. ASIACRYPT 2004*, LNCS 3329, pp.119–136, Springer-Verlag (2004).
 - 34) Algesheimer, J., Camenisch, J. and Shoup, V.: Efficient Computation Modulo a Shared Secret with Application to the Generation of Shared Safe-Prime Products, *Proc. CRYPTO 2002*, LNCS 2442, pp.417–432, Springer-Verlag (2002).
 - 35) Damgård, I., Fitzi, M., Kiltz, E., Nielsen, J.B. and Toft, T.: Unconditionally Secure Constant-Rounds Multi-Party Computation for Equality, Comparison Bits and Exponentiation, *Proc. TCC 2006*, LNCS 3876, pp.285–304, Springer-Verlag (2006).
 - 36) Nishide, T. and Ohta, K.: Multiparty Computation for Interval, Equality, and Comparison without Bit-Decomposition Protocol, *Proc. PKC 2007*, LNCS 4450, pp.343–360, Springer-Verlag (2007).
 - 37) Schoenmakers, B. and Tuyls, P.: Efficient Binary Conversion for Paillier Encrypted Values, *Proc. EUROCRYPT 2006*, LNCS 4004, pp.522–537, Springer-Verlag (2006).
 - 38) Rabin, M.O.: How to Exchange Secrets by Oblivious Transfer, Technical Report TR-81, Harvard University (1981).
 - 39) Kolesnikov, V. and Schneider, T.: Improved Garbled Circuit: Free XOR Gates and Applications, *Proc. ICALP 2008*, LNCS 5126, pp.486–498, Springer-Verlag (2008).
 - 40) Aumann, Y. and Lindell, Y.: Security against Covert Adversaries: Efficient Protocols for Realistic Adversaries, *Proc. TCC 2007*, LNCS 4392, pp.137–156, Springer-Verlag (2007).
 - 41) Lindell, Y. and Pinkas, B.: An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries, *Proc. EUROCRYPT 2007*, LNCS 4515, pp.52–78, Springer-Verlag (2007).
 - 42) Goyal, V., Mohassel, P. and Smith, A.: Efficient Two Party and Multi-Party Computation against Covert Adversaries, *Proc. Eurocrypt 2008*, LNCS 4965, pp.289–306, Springer-Verlag (2008).
 - 43) 情報処理学会コンピュータセキュリティ研究会(CSEC): コンピュータセキュリティシンポジウム 2009, 入手先 (<http://www.iwsec.org/css/2009/>) (参照 2011-03-17).
 - 44) Malkhi, D., Nisan, N. and Pinkas, B., et al.: FairplayMP, available from

(<http://www.cs.huji.ac.il/project/Fairplay/fairplayMP.html>) (accessed 2011-03-17).

- 45) VIFF Development Team: VIFF, the Virtual Ideal Functionality Framework, available from (<http://viff.dk/>) (accessed 2011-03-17).
- 46) SecureSCM Project: Welcome to SecureSCM Project, available from (<http://www.securescm.org/>) (accessed 2011-03-17).
- 47) Sharemind: News blog, available from (<http://research.cyber.ee/sharemind/>) (accessed 2011-03-17).
- 48) Malkhi, D., Nisan, N. and Pinkas, B., et al.: Fairplay, available from (<http://www.cs.huji.ac.il/project/Fairplay/>) (accessed 2011-03-17).
- 49) Seagate: Barracuda 7200.10, available from (http://www.seagate.com/docs/pdf/datasheet/disc/ds_barracuda_7200_10.pdf) (accessed 2011-03-17).

付 録

A.1 Free XOR による高効率化

本付録では, Free XOR と呼ばれる, 歪曲回路の生成および実行の計算コストや, 歪曲回路の通信コストを削減できる方式について説明し, 提案方式への適用について考察する. そしてその適用効果を定量的に評価する.

A.1.1 従来研究

Free XOR は, Kolesnikov ら³⁹⁾ によって提案され, 歪曲回路の生成および実行の計算コストや, 歪曲回路の通信コストを削減できる. 文献 19) では Free XOR を効果的に用いた実装報告が示されている. Free XOR の特徴は, 歪曲回路の生成において (W_i^0, W_i^1) を $(W_i^0, W_i^0 \oplus P)$ に置き換えたことである (P は κ ビットの乱数であり各ワイヤに共通の値). そして g が XOR ゲートであれば, 歪曲真理値表を用いず, 単に入力の歪曲値 $(W_i^b, b \oplus c_i)$, $(W_j^{b'}, b' \oplus c_j)$ の排他的論理和 $(W_i^b \oplus W_j^{b'}, b \oplus c_i \oplus b' \oplus c_j)$ を出力の歪曲値とする. すると前記の置き換えから

$$W_i^b \oplus W_j^{b'} = \begin{cases} W_i^0 \oplus W_j^0 & (b = b') \\ W_i^0 \oplus W_j^0 \oplus P & (b \neq b') \end{cases} \quad (3)$$

となることから, $W_k^0 = W_i^0 \oplus W_j^0$, $c_k = c_i \oplus c_j$ とすれば良い. その他のゲートについては文献 25) の方法と同様である.

文献 25) の方法に Free XOR を適用した場合, XOR の出力ワイヤには歪曲値を新たに割り当てる必要がなく, その他のワイヤについても歪曲値生成に必要な乱数は半分でよ

い. また歪曲回路のデータサイズや実行時の疑似ランダム関数計算についても XOR のゲート数に比例して削減できる.

A.1.2 提案方式への適用

提案方式は基本方式, 変形方式のいずれも, 回路の入力の歪曲値を Proxy 以外の主体が生成し, その他の歪曲値は Proxy が生成するため, 単純に考えれば共通乱数 P を少なくとも 2 主体が共有する必要があり, 入力の実匿性に影響する. しかし入力の歪曲値以外のみ共通乱数 P を用いることで, そのような共有は不要となる. 以下では基本方式を例に具体的な手続きを与える.

Client i の入力: $a_i \in \{0, 1\}^*$, PK_S (Server の公開鍵^{*1})

Proxy の入力: $P \in_R \{0, 1\}^\kappa$

Server の入力: SK_S (PK_S に対応する秘密鍵)

Server の出力: $f(a_1, \dots, a_N)$

- (1) Client i は入力 a_i の各ビット b について, 歪曲値 (W^0, c) , (W^1, \bar{c}) を生成し, $(W^b, b \oplus c)$ を PK_S で暗号化し, (W^0, c) , (W^1, \bar{c}) および $(W^b, b \oplus c)$ の暗号文 C をセキュア通信路より Proxy に送信する.
- (2) Proxy は以下の処理を行い歪曲回路を生成する.
 - (a) ビット b を入力するワイヤ w に対し, Client より得た (W^0, c) , (W^1, \bar{c}) から以下の 2 値をランダムな順序に並べたエントリを持つ変換表を作成し, 歪曲値を変換する.

$$\begin{cases} c : (W'^0, c') \oplus F_{W^0}(c) \\ \bar{c} : (W'^0 \oplus P, \bar{c}') \oplus F_{W^1}(\bar{c}) \end{cases} \quad (4)$$

ここで $W'^0 \in \{0, 1\}^\kappa$, $c' \in_R \{0, 1\}$ とする.

- (b) 論理回路 f のワイヤのうち, ビット b を入力としないすべてのワイヤ w_i に対し, $W_i^0 \in_R \{0, 1\}^\kappa$ を生成し, $W_i^1 = W_i^0 \oplus P$ を計算して (W_i^0, c_i) , (W_i^1, \bar{c}_i) をそれぞれワイヤ w_i を流れるビット 0, 1 に対応する歪曲値として割り当てる.
- (c) 入出力ワイヤ (w_i, w_j) , w_k を持つゲート g に対し, 式 (1) をランダムな順序に並べたエントリを持つ歪曲真理値表 T_g を割り当てる.
- (d) T_g で構成された歪曲回路および各 Client から受信した暗号文 C を Server に

*1 任意の公開鍵暗号アルゴリズムでよい.

表 8 XOR ゲートの比率 (100 入力)
Table 8 Ratio of XOR gates (100 inputs).

	XOR	全体	割合
5 ビット最大値	396	3,970	11.43%
10 ビット最大値	891	8,435	12.00%
20 ビット最大値	1881	17,365	12.26%
5 ビットクロス集計	25,152	51,054	50.17%

表 9 XOR ゲートの比率 (1,000 入力)
Table 9 Ratio of XOR gates (1,000 inputs).

	XOR	全体	割合
5 ビット最大値	3,996	39,970	11.43%
10 ビット最大値	8,991	84,935	12.00%
20 ビット最大値	18,981	174,865	12.26%
5 ビットクロス集計	255,328	515,816	50.03%

送信する .

- (3) Server は C を復号して回路への入力ビット b に対応する歪曲値 ($W^b, b \oplus c$) を取得し, ビット b を入力するワイヤ w が複数ある場合は, 2 つ目以降について式 (4) に基づき歪曲値を変換する . そして入出力ワイヤ (w_i, w_j), w_k を持つゲート g に割り当てられた歪曲真値表 T_g について繰り返し以下を行い, 最終的に計算結果 $f(a)$ を得る .

- (a) w_i, w_j に対応する歪曲値 ($W_i^b, b \oplus c_i$), ($W_j^{b'}, b' \oplus c_j$) を入力する .
- (b) T_g のうち, ($b \oplus c_i, b' \oplus c_j$) と一致するラベルのエントリを取り出し, 当該エントリのマスクを入力歪曲値から生成して打ち消すことで, w_k の歪曲値 ($W_k^{g(b,b')}, g(b, b') \oplus c_k$) を求め出力する .
- (c) 論理回路の出力を返すワイヤの歪曲値の最終ビット (歪曲値が ($W_k^{g(b,b')}, g(b, b') \oplus c_k$) であれば $g(b, b') \oplus c_k$) をあらかじめ定められた順序で結合したものを $f(a)$ とする .

A.1.3 評 価

表 2 ~ 表 5 で実行した回路について, XOR ゲートの比率を見積もった . 結果は表 8, 表 9, 表 10 のとおり, 最大値は全体の 1 割強だが, クロス集計はおよそ半分が XOR ゲートによって構成されていることが分かる . したがってクロス集計に関しては大きな改善効果が見

表 10 XOR ゲートの比率 (10,000 入力)
Table 10 Ratio of XOR gates (10,000 inputs).

	XOR	全体	割合
5 ビット最大値	39,996	399,970	11.43%
10 ビット最大値	89,991	849,935	12.00%
20 ビット最大値	189,981	1,749,865	12.26%
5 ビットクロス集計	2,559,200	5,122,980	50.00%

込める .

(平成 22 年 3 月 15 日受付)

(平成 23 年 3 月 7 日採録)



千田 浩司 (正会員)

平成 12 年早稲田大学大学院理工学研究科数理科学専攻修士課程修了 . 同年日本電信電話株式会社 (NTT) 入社 . 博士 (工学) . 現在, プライバシ保護技術, 暗号応用技術の研究開発に従事 . 平成 13 年電子情報通信学会 SCIS 論文賞受賞 . 電子情報通信学会会員 .



五十嵐 大 (正会員)

平成 20 年東京大学大学院情報理工学系研究科コンピュータ科学専攻博士前期課程修了 . 同年日本電信電話株式会社 (NTT) 入社 . 日本ソフトウェア科学会会員 .



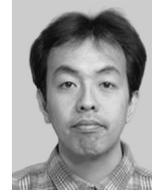
柴田 賢介 (正会員)

平成 13 年九州大学工学部電気情報工学科卒業 . 平成 15 年同大学大学院システム情報科学府修士課程修了 . 同年日本電信電話株式会社 NTT 情報流通プラットフォーム研究所に入所 . 以来, 情報セキュリティの研究に従事 . 平成 18 年第 61 回 GN 研究会優秀発表賞受賞 . 平成 19 年山下記念研究賞受賞 .



山本 太郎 (正会員)

平成 6 年北海道大学大学院工学研究科情報工学専攻修士課程修了。同年日本電信電話株式会社 (NTT) 入社。現在, ネットの安心等に関する社会科学的アプローチからの情報セキュリティ技術の研究開発に従事。日本社会情報学会 (JSIS) 会員。



高橋 克巳 (正会員)

昭和 63 年東京工業大学理学部数学科卒業。平成 18 年東京大学大学院情報理工学系電子情報学専攻博士課程修了。昭和 63 年日本電信電話 (株) 入社。以来, 情報検索, データマイニング, プライバシ保護データ処理, 暗号, 情報セキュリティ, セキュリティ社会科学の研究開発に従事。博士 (情報理工学)。平成 12 年度本会論文賞受賞。電子情報通信学会会員。