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

柴田 賢介† 千田 浩司† 五十嵐 大† 山本 太郎† 高橋 克巳†

† 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
180-8585 東京都武蔵野市緑町 3-9-11

shibata.kensuke@lab.ntt.co.jp

あらまし 近年，個人のプライバシー情報や企業の機密情報を保護しながら利活用するニーズが高まっている．この利活用を実現する技術の 1 つとして，秘密計算がある．本稿では，IWSEC2008 において著者らが提案した高速なセキュアマルチパーティプロトコルである委託型 2 パーティプロトコルをもとに，秘匿回路計算を行なうシステムを Microsoft Excel をフロントエンドとして実装した結果について報告する．特に最大値やクロス集計などの基本的な統計演算の性能評価を行ない，実装したソフトウェアの有用性を示す．

Delegated Two-party Secure Function Evaluation System with Spreadsheet Front-end

Kensuke Shibata† Koji Chida† Dai Ikarashi† Taro Yamamoto†
Katsumi Takahashi†

†NTT Information Sharing Platform Laboratories
3-9-11 Midori-Cho Musashino-Shi Tokyo 180-8585 Japan

shibata.kensuke@lab.ntt.co.jp

Abstract In recent years, there is a need for widespread utilization of privacy or confidential information as well as preserving one. One of the technologies which meets the need is secure function evaluation. In this paper, we show the design and implementation of delegated two-party secure function evaluation system with spreadsheet front-end. We evaluate our system and show the feasibility via statistical operation.

1 はじめに

近年のインターネットの普及に伴い，ネットワークを介した様々なサービスが提供されている．これらの中にはライフログとして蓄積された個人の購買履歴や行動履歴を活用してレコメンデーションを行なうといったものや，企業等において管理される情報を電子化し，これを収集／分析する，といったものなど，ネットワークを介して流通する情報を利活用するサービスがある．

従来，個人のプライバシー情報や企業等において管理される機密情報は厳重な保護の対象とされてきたが，これらの情報を保護しながら利活用することへの動きが促進されつつある．産業分野においては” 情報大航海プロジェクト”[1] において，情報利活用についての検討と実証事業が行なわれており，医療分野においては，” 健康情報活用基盤実証事業”[2] が平成 20 年度より行なわれている．

プライバシー情報や機密情報の利活用のためには，プライバシー保護や企業情報漏洩対策の観点から，これらの情報を保護しつつ利活用を可能とする技術が必要となる．現在，PPDA(Privacy Preserving

Data Analysis) のための主要な技術として，匿名化，再構築計算，秘密計算の 3 つの技術が検討されている．

本研究では，上記 3 種類のプライバシー保護データ処理技術のうち，生データを一切復元せずに各種演算を実行することを可能とする秘密計算技術に着目する．同技術は，個人情報や機密情報の生データを第三者に提供することなく，統計／分析を可能とする技術であり，個人情報の第三者提供に伴うリスクや，企業情報漏洩のリスクを低減しうるものである．

筆者らは，秘密計算の一方式として Yao が提案している 2 パーティ秘匿回路計算 [3] に対し，1-out-of-2 oblivious transfer プロトコル [4] を用いることなく秘匿回路計算を実現する方式を提案している [5]．本稿では，文献 [5] において提案した手法（以後 IWSEC2008 方式とする）に対し，仲介者を配置することなく秘匿回路計算を実現する方式を提案するとともに，表計算ソフトウェアをフロントエンドとして提案手法を実装した結果について報告する．実装したソフトウェアを用いて最大値やクロス集計などの基本的な統計演算の性能評価を行なうことにより，提案手法の有用性を示す．

2 既存研究

本章では，提案手法の元となる 2 パーティ秘匿回路計算 [3] と IWSEC2008 方式について概説する．

2.1 2 パーティ秘匿回路計算

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が乱数化し，もう一方のパーティ（秘匿回路計算サーバ）が乱数化された回路を用いて目的の演算を行なうプロトコルである．本プロトコルはクライアント－サーバ型であり，クライアントへ入力 α，サーバへは関数 f を入力とすると，クライアントは f に関する情報を，サーバは α に関する情報を得ることなく $f(\alpha)$ を求めることができる．関数 f は，論理ゲート $g : \{0,1\} \times \{0,1\} \rightarrow \{0,1\}$ を組み合わせた論理回路である．

以下に処理の例を示す．

1. サーバにおいて以下のステップで処理を行なう．
   (a) 回路の入力ワイヤ i に対し，2 つのランダム値 $W_i^0, W_i^1 \in \{0,1\}^\kappa$ を 0，1 に対応する値として割り当てる．κ はセキュリティパラメータである．
   (b) ワイヤ i のランダム値として $(< W_i^{0)}, c_i >, < W_i^1, \overline{c}_i >)$ を割り当てる．c_i はランダムビットであり，$\overline{c}_i = 1 - c_i$ を満たす．
   (c) 2 つの入力 i，j，出力 k を持つ論理ゲート 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 T_g を割り当て，真理値表を乱数化したテーブルを生成する．

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

   $F_W : \{0,1\}^{\kappa+1} \rightarrow \{0,1\}^{\kappa+1}$ は擬似ランダム関数である．g が論理回路の最終ゲートである場合，c_k に 0 をセットすることにより，回路の最終的な出力，すなわち $f(\alpha)$ のビット列を得ることが可能となる．
   (d) T_g で構成された乱数化回路をクライアントへ送付する．
2. ワイヤ i への入力となる，α の個々のビット b について，クライアント－サーバ間で 1-out-of-2 oblivious transfer を実行することにより，クライアントは $< W_i^b, b \oplus c_i >$ を得る．
3. クライアントは $< W_i^b, b \oplus c_i >$ を入力として乱数化回路を実行し，$f(\alpha)$ を得る．

これにより，入力ビット b，b' に対応する乱数値は $< W_i^b, b \oplus c_i >$，$< W_j^{b'}, b' \oplus c_j >$ と表され，T_g から $g(b, b')$ の乱数に対応させた $< W_k^{g(b,b')}, g(b, b') \oplus c_k >$ を得ることができる．これを α のすべてのビットに対して実行することにより，クライアントは $g(b, \overline{b}')$，$g(\overline{b}, b')$，$g(\overline{b}, \overline{b}')$，の結果を得ることなく，目的の演算結果を得ることが可能となる．

2.2 IWSEC2008 方式

筆者らは，前節で述べた Yao の 2 パーティ秘匿回路計算をベースとし，より簡易な手法を用いて秘

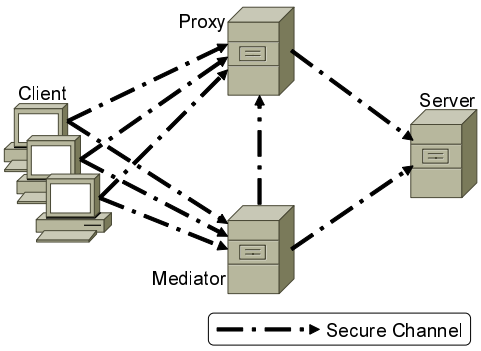

図 1: IWSEC2008 方式

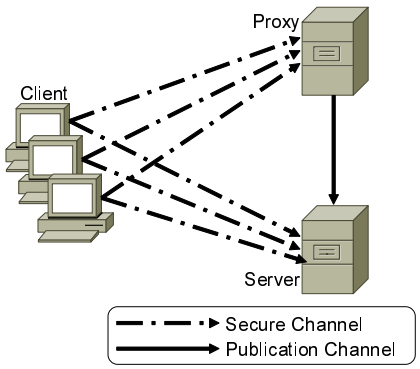

図 2: 提案手法

匿回路計算を実現するプロトコル（IWSEC2008 方式）を提案している．本稿で提案・実装する秘匿回路計算は，IWSEC2008 方式において Protocol として提案された方式を改良したものである．本節では，Protocol について概説する．

IWSEC2008 方式では，図 1 に示すように，秘匿回路計算を実行する 2 パーティ（Proxy，Server）に加えて，Clients からデータを預かり，秘匿処理を実施して Proxy，Server へデータを受け渡す Mediator が存在するモデルである．処理は以下の順で行なわれる．

1. Mediator は回路の入力ワイヤ i について，0 と 1 に対応する 2 つの乱数値 $W_i^0, W_i^1 \in \{0,1\}^\kappa$ を生成する．また，ランダムビット c_i を生成する．

2. 個々の ClientC_ν が保持する生データ $\alpha_\nu \in \{0,1\}^l$ を 2 つの断片 s_ν，t_ν に分割し，s_ν，t_ν をそれぞれ Proxy と Mediator へ送付する．なお，$s_\nu \in_R \{0,1\}^l$，$t_\nu = \alpha_\nu \oplus s_\nu$ である．

3. Proxy は s_ν を用いて関数 f に対応する乱数化回路 T_g を生成し，この乱数化回路を Server へ送信する．T_g の生成には前節において述べた Yao の方式を用いる．

4. Mediator は t_ν の個々のビット b に対応する $< W_i^b, b \oplus c_i >$ を Server へ送信する．

5. Server は T_g と $< W_i^b, b \oplus c_i >$ から $f(\alpha_1, ..., \alpha_n)$ を得る．

IWSEC2008 方式では，Yao の方式におけるクライアントの処理を Proxy において実行していると言える．

3 提案手法とその実装

本章では，IWSEC2008 方式をベースとし，Mediator を必要とすることなく秘匿回路計算を実行することが可能となる方式を提案し，提案手法の実装について述べる．

3.1 提案手法

IWSEC2008 方式では，秘匿回路計算を実行する 2 パーティ (Proxy，Server) に加えて，Client が保持するデータの受け渡しを仲介する Mediator が必要であった．本節では，Mediator において行なわれていた処理を Client において実行する方式を提案する．これにより，秘匿回路計算を実行する際に必要となるサーバの台数を減らすことができるとともに，Mediator を介して行なわれていたデータの授受を省略することで，通信のオーバヘッドの低減を見込める．

提案手法におけるプレイヤとデータの授受を図 2 に示す．プロトコルの流れは以下のとおりである．

1. Client において，回路の入力ワイヤ i について，0 と 1 に対応する 2 つの乱数値 $W_i^0, W_i^1 \in \{0,1\}^\kappa$ ならびにランダムビット c_i を生成する．

2. 個々の ClientC_ν が保持する生データ $\alpha_\nu \in \{0,1\}^l$ を 2 つの断片 s_ν，t_ν に分割し，s_ν を Proxy へ送付する．また，t_ν の個々のビット b に対応する $< W_i^b, b \oplus c_i >$ を Server へ送付する．なお，$s_\nu \in_R \{0,1\}^l$，$t_\nu = \alpha_\nu \oplus s_\nu$ である．

3. Proxy は s_ν を用いて関数 f に対応する乱数化回路 T_g を生成し，この乱数化回路を Server へ送信する．

4. Server は T_g と $< W_i^b, b \oplus c_i >$ から $f(\alpha_1, ..., \alpha_n)$ を得る．

提案手法では，IWSEC2008 方式において Mediator が実行していた乱数生成の処理を Client において実行することになるため，Client の処理が増加するが，提案手法の実装ではこれを表計算ソフト上で利用者が容易に実行可能とすることにより，利用者側への負担増加を抑制している．

なお，提案手法のセキュリティについては，文献 [5] において議論されている 2 パーティ秘匿回路計算において Mediator が行なっていた処理を Client において実行することとなるため，IWSEC2008 方式に比べると Client を攻撃者として想定しなければならないケースが増えることとなる．Client からサーバサイドへ不正な値を送りつける攻撃に関しては，Proxy，Server それぞれが不正な値の断片を用いて乱数化回路の生成および秘匿回路計算を行ない，計算結果 (不正な値を用いているため意味のない計算結果となる) を Client に返すこととなり，攻撃自体がほぼ意味を成さないこととなる．

3.2 2 パーティ秘匿回路計算の実装

本節では，提案手法に基づいて 2 パーティ秘匿回路計算を実行するシステムの実装について述べる．図 3 にシステム構成を示す．図 2 における Client での処理は，Microsoft Excel のプラグインとして実装しており，利用者が Excel 上において秘匿化したいデータを矩形選択し，メニュー上から「秘匿処理」を選んで実行する．その結果，秘匿化されたデータが新しいシートに出力される．複数の Client において秘匿化されたデータは，代表 Client が Excel 上で統合し，秘密計算の対象となるデータを矩形選択した上で，Excel のメニュー上から「秘密計算」を選んで実行する．この際に，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

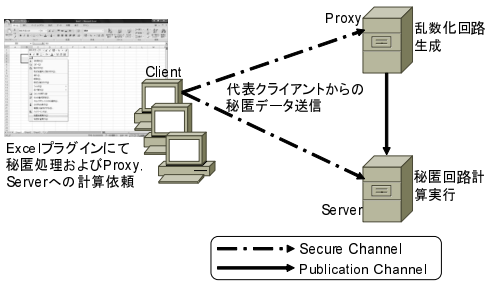

図 3: 秘匿回路計算システムのシステム構成

なお，秘匿化されたデータの統合に際しては，データの提供は行なわず，統合のみを実行する分析者を Client の 1 つとして配置することも可能である．

Proxy と 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 節において述べた通り，乱数化回路の生成と秘匿回路計算の実行を行ない，計算結果を Client へ返却する．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．

なお，図 2 において，Client から Proxy および Server へのデータの送信にはセキュアチャネルを用いているが，本実装においてはこれを PKI によって実現しており，Client は Proxy および Server の公開鍵を用いて送付するデータを暗号化した上でそれぞれに送信している．なお，暗号化の処理は上記 Excel プラグインでの「秘匿処理」の際に実行され，利用者には乱数値生成と暗号化を個別に実行する必要はない．

4 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なった結果を示す．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，およびクロス集計について，入力データの数とビット数をパラメータとし，秘匿回路計算に要する時間を計測した．評価結果を表 2〜表 6 に示す．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なった場合に要する時間，計算時間（2）は関数 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なっておいた場合に要する時間である．それぞれ 3 回の計測を行ない，平均値を実験結果としている．

クロス集計の計測に関しては，搭載メモリ量に限界があったため，入力データは 5 ビットのみとなっている．計算結果より，最大値計算およびクロス集

計ともに，入力データならびに入力ビット数に比例して計算時間が増大している．また，乱数生成，秘匿回路生成を事前に行なった場合，計算時間を50%程度削減できることが分かった．

なお，論理回路 1 ゲートあたりの計算時間については，乱数生成・秘匿回路生成を事前に行なわない場合で平均 11.14 μ sec（分散：0.3783），事前に行なった場合には平均 6.01 μ sec（分散：4.254）との結果が得られている．後者については分散の値が大きくなっているが，これは入力データ数が 100 の場合において，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．入力データ数 1,000 および 10,000 の場合のみを対象とした事前計算ありの場合の計算時間は，平均 4.79 μ sec （分散：0.0692）となっている．

表 1: 性能評価実験環境

CPU	Intel Core2 Quad Q9650(3.0GHz)
メモリ	4GB
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

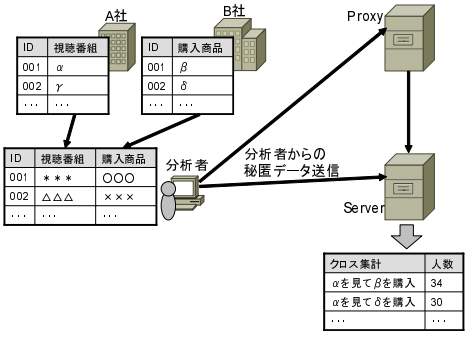

図 4: 秘匿回路計算適用事例

表 2: 実験結果（最大値）：入力ビット=5 ビット

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	4272.08
計算時間（2）[msec]	42.35	186.68	1811.97

表 3: 実験結果（最大値）：入力ビット=10 ビット

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	9116.12
計算時間（2）[msec]	87.89	418.05	3809.19

表 4: 実験結果（最大値）：入力ビット=15 ビット

入力データ数	100	1,000	10,000
ゲート数	12,900	129,900	1,299,900
回路の深さ	128	182	254
計算時間（1）[msec]	145.76	1376.17	13,894.53
計算時間（2）[msec]	107.29	615.62	5,903.68

表 5: 実験結果（最大値）：入力ビット=20 ビット

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 6: 実験結果（クロス）：入力ビット=5 ビット

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

5　秘匿回路計算の適用事例

前節において，実装した 2 パーティ秘匿回路計算の性能評価に関する結果を示した．提案手法により，論理回路 1 ゲートあたり平均 11.14 μ sec（事前に秘匿回路生成等を行なった場合には 4.79 μ sec）の性能で秘匿回路計算を実行することが可能となっており，秘匿回路計算を適用することが可能な分野は拡がっていると言える．本節では，秘匿回路計算を用いたクロス集計の適用分野について述べる．

図 4 に秘匿回路計算の適用例を示す．この例では，ポータルサイト上などで互いに同一のユーザを顧客としている動画配信サイト A 社とショッピングサイト B 社が，互いの視聴履歴および購買履歴を明かすことなく，クロス集計の結果のみを取得することが可能となる．

処理の手順は以下のとおりである．

1. A 社，B 社が提案手法における Client となり，今回実装した Excel のプラグインを用いて ID 以外のデータをそれぞれ秘匿化し，分析者へ送付する．
2. 分析者は A 社／B 社から送付されてきた秘匿化済みデータを名寄せし，Proxy および Server に対して名寄せしたデータを送付するとともに，クロス集計を要求する．
3. Proxy ／ Server において秘匿回路の生成および計算を実行する．

クロス集計の結果を得ることにより，例えば動画 α を視聴した人が商品 β を多く購入しているといった分析が可能となり，B 社においては販売している商品と相関の強い動画に対して積極的に広告を流す，といったことが可能となる．

但し，クロス集計を取った結果の値がごく少数である場合には，計算結果から相手の生データを推測することが可能となる．クロス集計の結果を開示する際には閾値を設け，集計の結果少数となったデータは破棄するといった手法が有効であると考えられる．

6　まとめと今後の課題

本研究では，プライバシー情報や企業における機密情報を保護しながら利活用することを目的とし，データを秘匿化したまま各種演算を実行することができる秘匿回路計算について，従来方式で必須であった仲介者を配置することなく実現する手法を提案するとともに，表計算ソフトウェアをフロントエンドとして提案手法を実装した．実装したソフトウェアを用いて最大値やクロス集計などの基本的な統計演算の性能評価を行ない，論理回路 1 ゲートあたり平均 11.14 μ sec の性能（事前に秘匿回路生成等を行なった場合には 4.79 μ sec）で秘匿回路計算が実行可能であることを示した．

今後の課題としては，まず秘匿回路計算の計算効率の向上が挙げられる．アプローチとしては，秘匿回路計算に最適化された論理回路を検討することにより，計算結果の取得に必要な論理回路のゲート数を減らすことによって，計算時間の低減を図る．また，5 章において秘匿回路計算の適用事例について述べたが，医療や教育など，センシティブな情報を取り扱う分野をはじめとして秘匿回路計算を適用可能な分野についても今後も検討を進める予定である．

参考文献

[1] 経済産業省, 情報大航海プロジェクト, `http://www.meti.go.jp/policy/it_policy/daikoukai/`

[2] 経済産業省, 健康情報活用基盤構築のための標準化及び実証事業, `http://microsite.accenture.com/meti/Pages/`

[3] A.C.Yao. How to generate and exchange secrets, *Proc. of FOCS '86*, pp.162–167, IEEE Press, 1986.

[4] Even, S., Goldreich, O., Lempel, A. A randomized protocol for signing contracts. *Communications of the ACM 28(6)*, pp.637–647 , 1985.

[5] Koji Chida, Katsumi Takahashi. Privacy Preserving Computations without Public Key Cryptographic Operation, *Proc. of The 3rd International Workshop on Security (IWSEC 2008)*, pp.184–200 , 2008.