隠し共有ストレージ機能を用いた 入出力のサイズを隠す二者間秘密計算の実現 (不) 可能性

品川 和雅 †§　縫田 光司 §　金山 直樹 †　西出 隆志 †　花岡 悟一郎 §
岡本 栄司 †

† 筑波大学
§ 産業技術総合研究所
shinagawa@cipher.risk.tsukuba.ac.jp

あらまし 通常，秘密計算の入出力長は公開情報として扱うが，入出力長を隠したい状況も存在する．2013 年に，Lindell，Nissim，Orlandi は，入力長に関する情報を全く漏らさない二者間秘密計算プロトコルの実現可能性／不可能性に関する結果を示した．本論文では，隠し共有ストレージという新しい機能を定義し，標準モデルでは実現不可能であるがこの機能が存在するならば実現可能になる関数が存在することを示す．また，隠し共有ストレージ機能を用いても，依然として計算できない関数が存在することも示す．我々の結果は，標準モデルにおいて実現不可能な関数たちについて，その実現不可能性の強さに差があることを明らかにする．

On the (Im)possibility of Size-Hiding Secure Two-Party Computation Using Hidden Shared Storage Functionality

Kazumasa Shinagawa†§　Koji Nuida§　Naoki Kanayama†　Takashi Nishide†
Goichiro Hanaoka§　Eiji Okamoto†

†University of Tsukuba
§National Institute of Advanced Industrial Science and Technology
shinagawa@cipher.risk.tsukuba.ac.jp

Abstract In secure computations, even the sizes of inputs may be considered to be confidential in some settings. In 2013, Lindell et al. showed examples of functions for which such "size-hiding" secure two-party computation is impossible. In this paper, we introduce a new ideal functionality called "hidden shared storage" and show that, assuming the new functionality, some of these impossible (in the standard model) functions turns to securely computable, while some other of these functions remains impossible. Our results may be interpreted as revealing the differences of "strength of impossibility" among these functions.

1　イントロダクション

多くの秘密計算プロトコルでは，各参加者の入力長は公開情報として扱われている．一方，応用によっては入力長それ自身が秘密情報となり得る場合が存在する [MRK03, IP07, ACT11, LNO13, COV15]．例えば，企業と警察がそれぞれ顧客リスト X と容疑者リスト Y について，両方のリストに含まれる人物のリスト $X \cap Y$ を秘密計算したいのだが，企業側は顧客数 $|X|$ も秘密情報と考えており，警察にもその情報を渡したくない状況が考えられる．[ACT11] は，ランダムオラクルモデルの下で，$|X|$ を隠したまま $X \cap Y$ を計算するプロトコルを提案した．それでは，$|X|, |Y|$ を両方隠したまま $X \cap Y$ を秘密計算することはできるだろうか．

Lindell, Nissim, Orlandi [LNO13] は，入力長に関する情報を漏らさない二者間秘密計算を次のように定式化した [1]．片方の入力長を隠す

[1] 実際の入力を (多項式サイズの) 最大入力長までパディングする素朴な解決法は，入力長の上限という情報が漏れる点と，短い入力でも長い入力と同じ実行コストがか

ことをクラス 1 のサイズ秘匿，両方の入力長を隠すことをクラス 2 のサイズ秘匿とし，さらに出力やそのサイズを誰が受け取るかという設定によって，サイズ秘匿のクラスを 1.a/b/c/d/e および 2.a/b/c の 8 つに分類した (詳細は 2.1 節参照)．この分類の下で，[LNO13] はクラス 1.a/c/e のサイズ秘匿では任意の関数 [2] を秘密計算できるが，それ以外のクラスのサイズ秘匿では計算不可能な関数が存在することを示した (表 1, 標準モデル, $\forall f$)．また，関数 $X \cap Y$ に対してクラス 2 のサイズ秘匿で秘密計算することは不可能であることを示した (表 1, 標準モデル, $X \cap Y$)．さらに，いくつかの関数に関して不可能性を示すことで，サイズ秘匿のクラス間の比較不可能性などが示されている (表 1, 標準モデル, vecxor, omprf)．

暗号分野では，標準モデルで実現不可能な技術について，何らかの理想機能 (例えばランダムオラクル) を仮定して実現する研究成果が多い．本稿で扱うサイズ秘匿秘密計算についても，それに適した理想機能を見出すことは有意義と考えられる．

1.1 貢献

本稿では，隠し共有ストレージ (Hidden Shared Storage: HSS) という新しい機能 $\mathcal{F}_{\mathsf{HSS}}$ を提案し，この機能を仮定した場合のサイズ秘匿の秘密計算の実現 (不) 可能性について調べる．$\mathcal{F}_{\mathsf{HSS}}$ は以下の要件を満たすものとする．

(i) 参加者 P_i から (SET, u, s) が送られると，アドレス u の場所に文字列 s を保管する．

(ii) 参加者 P_i から (REQ, u) が送られると，アドレス u に保管された文字列 s を P_i に送る．保管されていなければ P_i に $\perp$ を送る．

(iii) $\mathcal{F}_{\mathsf{HSS}}$ は上記以外の動作を一切行わない．特に，アクセス (SET や REQ) に関する履歴を一切持たない．

(iv) 参加者 P_i が $\mathcal{F}_{\mathsf{HSS}}$ へ複数回アクセスしたとき，アクセス内容だけでなくアクセス回数や通信量さえも秘匿されている．

(iii) の要件は，$\mathcal{F}_{\mathsf{HSS}}$ が「保管と返答はするが，他は何もしない信頼できる第三者」であることを意味する．(iv) の要件は，P_i がプロトコル中のあるラウンドで $\mathcal{F}_{\mathsf{HSS}}$ に複数回アクセスするとき，実際のアクセス回数を他の参加者に推測されないことを意味する [3]．$\mathcal{F}_{\mathsf{HSS}}$ は標準モデルにおいて実現することはできないが，既存のクラウドストレージサービス (Dropbox 等) を用いると同様の機能が得られることが期待される．本稿では，$\mathcal{F}_{\mathsf{HSS}}$ を利用できる設定を，隠し共有ストレージモデル (HSS モデル) と呼び，利用できない設定を，標準モデルと呼ぶ．また，関数 $f(x, y)$ についてクラス A.B のサイズ秘匿をする秘密計算を，$(A.B, f(x, y))$ と表すことにする．我々は，HSS モデルにおける実現 (不) 可能性に関して，以下の結果を示した．

表 1: サイズ秘匿の実現 (不) 可能性

	$\forall f$	$X \cap Y$	vecxor	omprf
○ 標準モデル				
1.a/c/e	✓	✓	✓	✓
1.b	×	✓	✓	✓
1.d	×	✓	✓	×
2.a/c	×	×	✓	✓
2.b	×	×	×	✓
○ HSS モデル				
1.a/c/e	✓	✓	✓	✓
1.b	✓	✓	✓	✓
1.d	×	✓	✓	×
2.a/c	×	✓	✓	✓
2.b	×	✓	✓	✓

(1) 標準モデルではある関数 f に対して $(1.b, f)$ が実現不可能であることが知られていたが，HSS モデルでは任意の関数 f に対して $(1.b, f)$ が実現可能である．(表 1, HSS モデル, $\forall f$)

(2) ある関数の集合 $\mathcal{L}$ が存在して，$f \in \mathcal{L}$ に対して，標準モデルでは $(2.a/b/c, f)$ が実現不可能だが，HSS モデルでは $(2.a/b/c, f)$ が実現可能である．特に，$X \cap Y \in \mathcal{L}$ である．(表 1, HSS モデル, $X \cap Y$ と vecxor)

(3) 標準モデル・HSS モデルのどちらにおいても $(1.d, f)$ が実現不可能である関数 f が存在する．(表 1, HSS モデル, omprf)

(4) 標準モデル・HSS モデルのどちらにおいても $(2.a/b/c, f)$ が実現不可能である関数 f が存在する．(表 1, HSS モデル, $\forall f$)

かる点が望ましくない．[LNO13] や本稿のプロトコルでは，入力長はいかなる多項式にもバウンドされない．

[2] 本稿で扱う全ての関数は，多項式時間計算可能関数であるため，例えば “任意の関数” という表現は，“任意の多項式時間計算可能関数” を表すものとする．

[3] 各参加者が常時 $\mathcal{F}_{\mathsf{HSS}}$ へアクセスする等の方法を用いることで，近似的に実現可能であると考えられる．

標準モデルでサイズ秘匿を実現不可能な関数たちの間に，例えば $(2.a/b/c, X \cap Y)$ は HSS モデルでは実現可能だが $(1.d, \mathsf{omprf})$ は HSS モデルでも実現不可能である，といった新たな不可能性の階層の存在を明らかにできた．

2 章では，サイズ秘匿のクラス (2.1 節)，多項式時間プロトコル (2.2 節)，プロトコル中に用いる構成要素 (2.3 節) について説明する．3 章では，理想機能 $\mathcal{F}_{\mathsf{HSS}}$ の定義 (3.1 節) と，HSS モデルにおける安全性定義 (3.2 節) を行う．4 章では，(1) の結果を示すため紛失切り詰め通信というプロトコルを構成し (4.1 節)，これを用いて任意の関数 f に対して $(1.b, f)$ を実現するプロトコルを構成する (4.2 節)．さらに，HSS モデルの下で $(2.a/b/c, X \cap Y)$ を実現するプロトコルを構成することにより，(2) の結果を示す (4.3 節)．5 章では，(3) の結果を示すために，HSS モデルの下で $(1.d, \mathsf{omprf})$ の実現不可能性を証明し (5.1 節)，(4) の結果を示すために，ある性質を持つ f に対して HSS モデルの下で $(2.a/b/c, f)$ の実現不可能性を証明する (5.2 節)．

2 準備

集合 $\{u_1, u_2, \cdots, u_n\}$ を $\{u_i\}_{i=1}^n$ と表記する．集合 $\{1, 2, \cdots, n\}$ を $[n]$ と表記する．文字列 α, β を結合した文字列を $\alpha||\beta$ と表記する．

2.1 サイズ秘匿のクラス

Lindell ら [LNO13] は，二者間秘密計算におけるサイズ秘匿のクラスを次のようなクラスとサブクラスに分類した．ただし，参加者を P_1, P_2，それぞれの入力を x, y，秘密計算する関数を $f(x, y)$ とする．P_1, P_2 がそれぞれ異なる関数 $f_1(x, y), f_2(x, y)$ を出力として受け取る場合は考慮されていない．

クラス 1 片方の参加者の入力長を隠す．

(a) P_1 は $f(x, y)$ のみを，P_2 は $f(x, y)$ と $1^{|x|}$ のみを受け取る．

(b) P_1 は $f(x, y)$ のみを，P_2 は $1^{|x|}$ のみを受け取る．

(c) P_1 は $f(x, y)$ のみを，P_2 は $1^{|f(x,y)|}$ と $1^{|x|}$ のみを受け取る．

(d) P_1 は何も受け取らず，P_2 は $f(x, y)$ と $1^{|x|}$ のみを受け取る．

(e) P_1 は $1^{|f(x,y)|}$ のみを，P_2 は $f(x, y)$ と $1^{|x|}$ のみを受け取る．

クラス 2 両方の参加者の入力長を隠す．

(a) P_1 と P_2 はそれぞれ $f(x, y)$ のみを受け取る．

(b) P_1 は $f(x, y)$ のみを受け取り，P_2 は何も受け取らない．

(c) P_1 は $f(x, y)$ のみを，P_2 は $1^{|f(x,y)|}$ のみを受け取る．

2.2 多項式時間プロトコル

クラス A.B のプロトコル π に対して，参加者 P_1, P_2 がプロトコルの出力として受け取る情報をそれぞれ $\mathrm{OUT}_1^{A.B}(x, y), \mathrm{OUT}_2^{A.B}(x, y)$ と表記する．また，上記の対を $\mathrm{OUT}^{A.B}(x, y)$ と表記する．クラス 1.b の場合，$\mathrm{OUT}^{1.b} = (f(x, y), 1^{|x|})$ である．また，参加者 P_1, P_2 がプロトコルに要する実行時間をそれぞれ $\mathrm{TIME}_{P_1}^{\pi}, \mathrm{TIME}_{P_2}^{\pi}$ と表記する．

定義 2.1 (多項式時間プロトコル [LNO13])．π を二者間プロトコルとする．π がクラス $A.B$ の多項式時間プロトコルであるとは，ある多項式 p が存在して，任意の自然数 $\kappa \in \mathbb{N}$ を満たす任意のペア $x_1, x_2 \in \{0,1\}^*$ に対して，以下の式が $i \in \{1, 2\}$ について成り立つことである．

$$\mathrm{TIME}_{P_i}^{\pi}(x_1, x_2, \kappa) \leq p(|x_i| + |\mathrm{OUT}_i^{A.B}(x_1, x_2)| + \kappa)$$

2.3 構成要素

定義 2.2 (紛失通信の機能)．送信者 S と受信者 R に対して，紛失通信の理想機能 $\mathcal{F}_{\mathsf{OT}}$ は次のように動作する．

1. $(\mathsf{sender}, (x_0, x_1))$ を S から受け取ると，(x_0, x_1) を記録し，S と R の両方に *"Accept"* を送る．
2. $(\mathsf{reciever}, b \in \{0, 1\})$ を R から受け取ると，x_b を R に送り，S に *"Accept"* を送る．

定義 2.3 (回路秘匿性完全準同型暗号)．アルゴリズムの組 $(\mathsf{G}, \mathsf{E}, \mathsf{D}, \mathsf{Ev})$ が回路秘匿性完全準同型暗号 *(*回路秘匿 *FHE)* であるとは，$(\mathsf{G}, \mathsf{E}, \mathsf{D})$ が *IND-CPA* 安全な公開鍵暗号であり，任意の鍵 $(pk, sk) \leftarrow \mathsf{G}(1^\kappa)$，任意の平文 $m_1, \cdots, m_n$，任意の多項式サイズ回路 $\mathcal{C}$ に対して以下の条件を満たすときにいう．

- 以下の確率が無視できること．

$$\Pr\big[\mathsf{D}_{sk}(\mathsf{Ev}_{pk}(\mathcal{C}, \langle \mathsf{E}_{pk}(m_1), \cdots, \mathsf{E}_{pk}(m_n)\rangle)) \neq \mathcal{C}(m_1, \cdots, m_n)\big]$$

- 分布 X, Y が計算量的に識別できないこと．

$$X = \{\mathsf{Ev}_{pk}(\mathcal{C}, \langle \mathsf{E}_{pk}(m_1), \cdots, \mathsf{E}_{pk}(m_n)\rangle)\}$$
$$Y = \{\mathsf{E}_{pk}(\mathcal{C}(m_1, \cdots, m_n))\}$$

3 隠し共有ストレージモデル

この章では，隠し共有ストレージモデルという新しいモデルを導入する．3.1 節では，理想機能 $\mathcal{F}_{\mathsf{HSS}}$ を定義する．3.2 節では，HSS モデルにおける安全性定義を行う．

3.1 隠し共有ストレージの定義

定義 3.1 (HSS の機能)．*HSS* の機能 $\mathcal{F}_{\mathsf{HSS}}$ は，リスト L*(プロトコルの開始時は空のリスト)* を保持し，次のような動作を行う．

- P_i からの入力 $(\mathsf{SET}, u \in \{0,1\}^*, s \in \{0,1\}^*)$ に対して，(u, s) をリスト L に追加する．もし $(u, s') \in L$ という s' が既に存在するならば，追加する前に (u, s') は L から削除する．
- P_i からの入力 $(\mathsf{REQ}, u \in \{0,1\}^*)$ に対して，$(u, s) \in L$ という s が存在するならば P_i に s を送り，存在しないならば P_i に $\perp$ を送る．

上記の定義において，攻撃者 $\mathcal{A}$ には一切の情報が送られていないことに注意する．したがって，$\mathcal{A}$ は P_i を corrupt しない限り，通信事実さえも知ることができない [4]．

全ての参加者 P_i が $\mathcal{F}_{\mathsf{HSS}}$ に自由にアクセスできる設定を，隠し共有ストレージモデル (HSS モデル) と呼ぶ．その際，1 回の SET または REQ を 1 回の演算とみなす．SET で保存された (u, s) に対して，アドレス長 $|u|$ がセキュリティパラメータ程度の長さならば，u を知らない多項式時間アルゴリズムが s の場所を探し当てる確率は無視できる．

3.2 安全性定義

理想機能 $\mathcal{F}_{\mathsf{HSS}}$ にアクセスできる参加者 P_1, P_2 が，プロトコル π を実行したとする．P_1 の view の確率分布とは，P_1 が π の実行によって知り得る全ての情報に関する確率分布であり，以下の確率変数の組である：P_1 の入出力，$P_2/\mathcal{F}_{\mathsf{HSS}}$ から送られたメッセージ，$P_2/\mathcal{F}_{\mathsf{HSS}}$ に送ったメッセージ，P_1 が用いた乱数表．P_2 についても同様である．プロトコル π が HSS モデルにおいて安全であるとは，P_i の入出力を受け取り，P_i の view を作るシミュレータが存在することである．以下の安全性定義は，式としては [LNO13] と同一だが，P_i の view の意味が異なること ($\mathcal{F}_{\mathsf{HSS}}$ の有無) に注意する．

定義 3.2 (クラス A.B の安全性定義)．プロトコル π が関数 f をクラス $A.B$ で *semi-honest* 攻撃者に対して安全に計算するとは，確率的多項式時間アルゴリズム $\mathcal{S}_i (i = 1, 2)$ が存在して，全ての多項式のペア $q_1(\cdot), q_2(\cdot)$ に対して以下の *2* つの分布が計算量的に識別できないことである．

$$\left\{\left(\mathcal{S}_i(x_i, \mathrm{OUT}_i^{A.b}(x_1, x_2)), \mathrm{OUT}^{A.b}(x_1, x_2)\right)\right\}_{\kappa, x_1, x_2}$$
$$\left\{\left(\mathrm{view}_i^{\pi}(x_1, x_2, \kappa), \mathrm{OUT}^{A.b}(x_1, x_2)\right)\right\}_{\kappa, x_1, x_2}$$

ここで，$\kappa \in \mathbb{N}, x_1 \in \{0,1\}^{q_1(\kappa)}, x_2 \in \{0,1\}^{q_2(\kappa)}$ である．上記を満たす π を，秘密計算 $(A.B, f)$ を実現するプロトコルであると言う．

4 実現可能性に関する結果

この章では，HSS モデルにおける実現可能性に関する結果を示す．4.1 節では，紛失切り詰め通信という新しいプロトコルを定義し，HSS モデルにおいて実現可能であることを示す．4.2 節では，紛失切り詰め通信を用いて，任意の関数 f に対して $(1.b, f)$ を安全に計算するプロトコルを構成する．4.3 節では，標準モデルでは実現不可能だが HSS モデルでは実現可能である，クラス 2 のプロトコルについて考察する．

4.1 紛失切り詰め通信

本節では，紛失切り詰め通信 (Oblivious Truncated Transfer: OTT) という新しい二者間プロトコルを定義し，HSS モデルの下で構成する．送信者の入力は n 個のデータ $x_1, x_2, \cdots, x_n$ であり，受信者の入力はリクエスト個数 m である．$m \leq n$ ならば受信者は $x_1, x_2, \cdots, x_m$ を受け取り，$m > n$ ならば受信者は $x_1, x_2, \cdots, x_n$ を受け取る．安全性の直観は，送信者が m を推測できないことと，$m \leq n$ のときに受信者が n および $x_{m+1}, x_{m+2}, \cdots, x_n$ に関して出力以上の情報を得られないことである．OTT の理想機能を以下のように定義する．

[4] 我々のプロトコルでは通信事実そのものを隠す必要はなく，同じラウンドにおける通信回数を隠せればよい．今回は簡単のためこのように定義する．

定義 4.1 (OTT の理想機能). $\mathcal{F}_{\mathsf{OTT}}$ を *OTT* の理想機能とする．送信者 S と受信者 R に対して $\mathcal{F}_{\mathsf{OTT}}$ は次のように動作する．

1. $(\mathsf{sender}, (x_1, x_2, \cdots, x_n))$ を S から受け取ると，$(x_1, x_2, \cdots, x_n)$ を記録し，S と R の両方に *"Accept"* を送る．ここで，$x_i \in \{0,1\}^*$ のビット長はそれぞれ異なってもよい．
2. $(\mathsf{reciever}, m)$ を R から受け取ると，$m \leq n$ ならば $(x_1, x_2, \cdots, x_m)$ を R に送り，$m > n$ ならば $(x_1, x_2, \cdots, x_n)$ を R に送る．S に *"Accept"* を送る．

送信者 S (入力 $x_1, x_2, \cdots, x_n$) と受信者 R (入力 m) による semi-honest 攻撃者に対して安全な OTT を，HSS モデルの下で構成する [5]．用いる要素技術は，IND-CPA 安全な公開鍵暗号 $(\mathsf{G}, \mathsf{E}, \mathsf{D})$ および semi-honest 攻撃者に対して安全な紛失通信プロトコルである．

紛失切り詰め通信 (OTT) プロトコル

1. $\ell := \lfloor \log_2 n \rfloor$ とする．S は $\mathsf{G}(1^\kappa)$ を ℓ 回実行して $\{(pk_j, sk_j)\}_{j=1}^{\ell}$ を作る．次に，S は $1 + \sum_{j=1}^{\ell+1} \lfloor n/2^{j-1} \rfloor$ 個の κ ビット乱数列を生成する．ここで，乱数列は 0^κ 以外からそれぞれ異なるように選ぶ．この乱数列たちを $\ell + 2$ 個のグループに分け，以下のように名前をつける．

$$\{u_i^{(1)}\}_{i=1}^{n}, \{u_i^{(2)}\}_{i=1}^{\lfloor n/2 \rfloor}, \cdots, \{u_i^{(j)}\}_{i=1}^{\lfloor n/2^{j-1} \rfloor}, \cdots$$
$$\cdots, \{u_i^{(\ell)}\}_{i=1}^{\lfloor n/2^{\ell-1} \rfloor}, \{u_1^{(\ell+1)}\}, \{u^{(\infty)}\}.$$

2. 全ての $i \in [n]$ について，S は $(\mathsf{SET}, u_i^{(1)}, x_i)$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力する．
3. 全ての $j \in [\ell], i \in [\lfloor n/2^j \rfloor]$ について，S は $\mathcal{F}_{\mathsf{HSS}}$ に $(\mathsf{SET}, u_i^{(j+1)}, u_{2i-1}^{(j)} || u_{2i}^{(j)} || E_j(u_{2i+1}^{(j)}))$ を入力する．もしも対応する $u_{2i+1}^{(j)}$ が存在しない場合，$E_j(0^\kappa)$ を用いる．ただし，$E_j(u)$ は u の公開鍵 pk_j についての暗号文である．
4. $0 \leq a < 2^\kappa$ をゼロ詰めにより κ 桁で表現したビット列を $(a)_\kappa$ と表記する．S は，以下の x に対して $(\mathsf{SET}, u^{(\infty)}, x || x_1 || x_2 || \cdots || x_n)$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力する．

$$x := (n)_\kappa || (|x_1|)_\kappa || (|x_2|)_\kappa || \cdots || (|x_n|)_\kappa$$

5. 全ての $j \in [\kappa]$ について，S と R は紛失通信を行う．ここで，m を κ 桁の二進表記したときの j 桁目を $m_j \in \{0,1\}$ とする (最下位を 1 桁とし，κ 桁までゼロ詰めする)．紛失通信の R の入力は m_j，S の入力は以下の $y_0^{(j)}, y_1^{(j)}$ である．
   - $j \leq \ell$ のとき：$y_1^{(j)} = 0 || u_1^{(j)} || sk_j$
   - $j > \ell$ のとき：$y_1^{(j)} = 1 || u^{(\infty)} || 0^{|sk_j|}$

   上記のいずれの場合も $y_0^{(j)} = 0^{|y_1^{(j)}|}$ とする．
6. m によって定まる集合 $J = \{j | m_j = 1\}$ の要素について，大きい順に並べたものを $j_1, j_2, \cdots$ とする．(先ほどの紛失通信によって，R は $\{y_1^{(j)}\}_{j \in J}$ を得ている．$y_1^{(j)}$ の先頭が 0 なら，$y_1^{(j)}$ には $u_1^{(j)}$ と sk_j が含まれており，先頭が 1 なら $u^{(\infty)}$ が含まれている．) もし $u^{(\infty)}$ を持っているなら，R は $(\mathsf{REQ}, u^{(\infty)})$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力し，返ってきた文字列から $x_1, \cdots, x_n$ を取り出し，ステップ 6 を終了する．もし $u^{(\infty)}$ を持っていないなら，R は $(\mathsf{REQ}, u_1^{(j_1)})$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力し文字列 s_1 を受け取る．$j_1 = 1$ ならば s_1 には x_1 が含まれており，$j_1 > 1$ ならば s_1 には $u_1^{(j_1-1)}, u_2^{(j_1-1)}$ および $E_{j_1-1}(u_3^{(j_1-1)})$ が含まれている．後者の場合，$(\mathsf{REQ}, u_1^{(j_1-1)})$ と $(\mathsf{REQ}, u_2^{(j_1-1)})$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力して次の文字列を受け取る．以下同様に繰り返して，最終的に $x_1, \cdots, x_{2^{j_1}}$ が得られる．次に，R は sk_{j_2} を保持しているので，上記の過程で $(\mathsf{REQ}, u_{2^{j_1-j_2-1}}^{(j_2+1)})$ によって得た暗号文 $E_{j_2}(*)$ を復号できる．復号結果が 0^κ ならば，ステップ 6 を終了する．そうでなければ，復号結果は $u_{2^{j_1-j_2}+1}^{(j_2)}$ であるため，$(\mathsf{REQ}, u_{2^{j_1-j_2}+1}^{(j_2)})$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力して文字列を受け取ることができる．以下，上記と同様の手順により $x_{2^{j_1}+1}, \cdots, x_{2^{j_1}+2^{j_2}}$ が得られる．上記の操作を $j_3, j_4, \cdots$ についても順に繰り返す．
7. R は，ステップ 6 で得られた $x_1, \cdots, x_m$ を出力する．($\sum_{j \in J} 2^j = m$ である．)

例 4.1. *S* の入力が $x_1, \cdots, x_7$ のときの *HSS* の構成を図 *1* に載せる *(*$u^{(\infty)}$ のページは省略してある*)*．図中の矢印は，始点のページには終点のページのアドレスが書き込まれているため，始点のページのアドレスを得られると，終点のページにもアクセスできることを表す．

定理 4.1. *IND-CPA* 安全な公開鍵暗号と *semi-honest* 攻撃者に対して安全な紛失通信が存在

[5] OTT を用いることで標準モデルでは不可能性が示されている関数を計算できることから (4.2 節)，OTT は標準モデルでは実現できない．

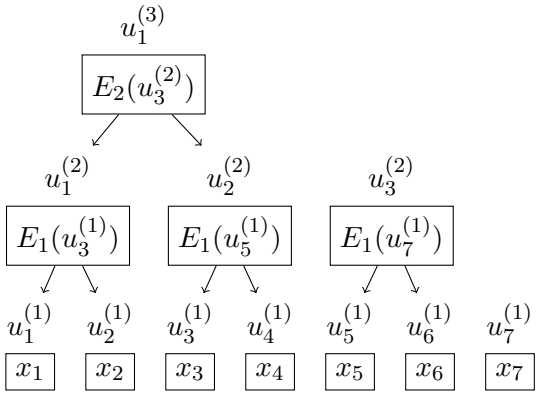

図 1: HSS の構成 ($n = 7$)

するならば，*HSS* モデルの下で $\mathcal{F}_{\mathsf{OTT}}$ を *semi-honest* 攻撃者に対して安全に実現できる．

証明. 紙幅の都合上，証明の概略を述べる．安全性を証明するためには，S, R の view を作り出すシミュレータ $\mathcal{S}_S, \mathcal{S}_R$ を構成すればよい．$\mathcal{S}_S$ の構成は容易である．なぜなら，紛失通信を除けば R は S にデータを送らないからである．$\mathcal{S}_R$ の構成は，$m > n$ と $m \leq n$ の 2 つに場合分けして考える．$m > n$ のとき，$\mathcal{S}_R$ は $(m, (x_1, \cdots, x_n))$ を受け取り，R の view を作るが，S の秘密情報 $(x_1, \cdots, x_n)$ を全て知っているため，シミュレートは容易である．$m \leq n$ のとき，$\mathcal{S}_R$ は $(m, (x_1, \cdots, x_m))$ を受け取り，R の view を作る．$\mathcal{S}_R$ は，$(x_1, \cdots, x_m)$ を入力とする OTT の送信者 S' になったつもりでステップ 1, 2, 3 で送信者が行う計算をそのまま行うことで，S から $\mathcal{F}_{\mathsf{HSS}}$ 経由で送られてくるデータをシミュレートできる．ここで，上記のシミュレーションで現実のプロトコルと異なる可能性がある部分は，暗号文の平文 (R はその秘密鍵を知らない) だけであることに注意する．以上のように $\mathcal{S}_S, \mathcal{S}_R$ を構成できるため，上記の OTT プロトコルは semi-honest 攻撃者に対して安全である． □

落とし戸付き一方向性置換族が存在するならば，semi-honest 攻撃者に対して安全な紛失通信および IND-CPA 安全な公開鍵暗号が存在するので，以下が成り立つ．

系 4.1. 落とし戸付き一方向性置換族が存在するならば，*HSS* モデルの下で $\mathcal{F}_{\mathsf{OTT}}$ を *semi-honest* 攻撃者に対して安全に実現できる．

4.2 クラス 1.b のプロトコル ($\forall f(x, y)$)

本節では，回路秘匿 FHE $(\mathsf{G}, \mathsf{E}, \mathsf{D}, \mathsf{Ev})$ および前節で構成した OTT を用いて，HSS モデルにおいては任意の関数 f に対して $(1.b, f)$ が semi-honest 攻撃者に対して安全に計算可能であることを示す．P_1 が保持する入力は $x_1 x_2 \cdots x_n$ のようにビット列で表されているものとする．回路 $C_{f,y}$ は n ビットの x を入力として受け取り，$f(x, y)$ のビット列 $w_1 w_2 \cdots w_t$ を出力する回路である．ここで $t = \max_{|x'|=n} |f(x', y)|$ 桁であり，実際のデータは $w_1 w_2 \cdots w_{|f(x,y)|}$ に含まれており，$|f(x, y)| + 1$ 桁以降は全てゼロである．回路 $C_{size,y}$ は n ビットの x を入力として受け取り，$|f(x, y)|$ のビット列を出力する回路である．ただし出力の桁数は κ 桁であり，上位ビットはゼロ詰めされているものとする．

クラス 1.b の $f(x, y)$ に対するプロトコル

1. P_1 は $(pk, sk) \leftarrow \mathsf{G}(1^\kappa)$ および n 個の暗号文 $c_i = \mathsf{E}_{pk}(x_i)$ を計算し，$(pk, c_1, \cdots, c_n)$ を P_2 に送る．
2. P_2 は $(pk, c_1, \cdots, c_n)$ を受け取り，
$$z \leftarrow \mathsf{Ev}_{pk}(C_{f,y}, (c_1, \cdots, c_n))$$
$$c_{size} \leftarrow \mathsf{Ev}_{pk}(C_{size,y}(\cdot), (c_1, \cdots, c_n))$$
を計算する．P_2 は P_1 に c_{size} を送る．
3. P_1 は $\ell \leftarrow \mathsf{D}_{sk}(c_{size})$ を計算する．
4. $L = \max_{|x'|=n} |f(x', y)|$ とする．P_2 を送信者 (入力 $z = (z_1, \cdots, z_L)$)，P_1 を受信者 (入力 ℓ) として，紛失切り詰め通信を行う．
5. P_1 は紛失切り詰め通信によって受け取った $(z_1, \cdots, z_\ell)$ をそれぞれ $w_i = \mathsf{D}_{sk}(z_i)$ によって復号し，$w = w_1 w_2 \cdots w_\ell$ を出力する．P_2 は何も出力しない．

上記のプロトコルが正しく $f(x, y)$ を計算することは，回路秘匿 FHE の性質から明らかである．

定理 4.2. もし回路秘匿 *FHE* および *OTT* が存在するならば，任意の関数 f に対して $(1.b, f)$ を *semi-honest* 攻撃者に対して安全に計算するプロトコルが存在する．

証明. 紙幅の都合上，証明の概略を述べる．安全性を証明するためには，P_1, P_2 の view を作り出すシミュレータ $\mathcal{S}_1, \mathcal{S}_2$ を構成すればよい．シミュレータ $\mathcal{S}_1$ は $(x, f(x, y))$ を受け取り，現実世界では P_2 から送られてくる c_{size} および $(z_1, \cdots, z_\ell)$ を作らなければならないが，$\mathcal{S}_1$ は $f(x, y)$ と $|f(x, y)|$ を知っているので，それらのビット列表示の各桁を自分で暗号化すればよい．シミュレータ $\mathcal{S}_2$ は $(y, 1^n)$ を受け取り，現実世界では P_1 から送られてくる $(pk, c_1, \cdots, c_n)$ を作

らなければならないが，FHE の IND-CPA 安全性より，自分で公開鍵とランダムな暗号文 n 個を生成すれば充分である．以上のように $\mathcal{S}_1, \mathcal{S}_2$ を構成できるので，上記のプロトコルは semi-honest 攻撃者に対して安全である． □

FHE が存在すれば，公開鍵暗号および semi-honest 攻撃者に対して安全な紛失通信を構成できる．この事実と前節の結果を組み合わせれば，以下の系が成り立つ．

系 4.2. もし回路秘匿 *FHE* が存在するならば，任意の関数 $f(x,y)$ に対して *HSS* モデルにおいて $(1.b, f(x,y))$ を *semi-honest* 攻撃者に対して安全に計算するプロトコルが存在する．

4.3 クラス 2 のプロトコル $(X \cap Y)$

集合 X と Y に対して，$X \cap Y$ を求める秘密計算は，サイズ秘匿の重要な応用である．しかしながら，クラス 2 では以下のような不可能性が示されている ([LNO13] の定理 5.4)．

定理 4.3 ($X \cap Y$ の実現不可能性 [LNO13])．標準モデルの下では，$2.a/b/c$ のいずれのクラスでも $X \cap Y$ の秘密計算は実現不可能である．

一方，HSS モデルの下では不可能性を覆すことができる．以下の定理が成り立つ．

定理 4.4. *HSS* モデルの下では，クラス $2.a/b/c$ で $X \cap Y$ の秘密計算を実現可能である．

証明. 任意の f に対して，$(2.b, f)$ のプロトコルを構成できれば $(2.a/c, f)$ をただちに実現できる．そのため，以下で $(2.b, X \cap Y)$ のプロトコルを構成する．

クラス 2.b の $X \cap Y$ プロトコル

1. 全ての $i \in [m]$ について，P_2 は $(\mathsf{SET}, y_i, 1)$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力する．
2. P_2 は P_1 に文字列 “OK” を送る．
3. 全ての $i \in [n]$ について，P_1 は (REQ, x_i) を $\mathcal{F}_{\mathsf{HSS}}$ に入力し，文字列 s_{x_i} を受け取る．受け取った文字列の集合を S として，P_1 は集合 $\{x_i | s_{x_i} \in S, s_{x_i} = 1\}$ を出力する．

上記のプロトコルに対してシミュレータ $\mathcal{S}_1, \mathcal{S}_2$ を構成することで安全性の証明をする．$\mathcal{S}_1$ は $X, X \cap Y$ を受け取り，P_1 の view を作る．非自明な部分はストレージ内容の再現であるが，全ての $x \in X \cap Y$ に対して $(\mathsf{SET}, x, 1)$ を $\mathcal{F}_{\mathsf{HSS}}$ に入力することで，ストレージを再現することができる．P_2 は一切のメッセージを受け取らないので，$\mathcal{S}_2$ は簡単に構成できる．したがって，上記の $X \cap Y$ プロトコルは semi-honest 攻撃者に対して安全である． □

この定理からただちに以下が成り立つ．

系 4.3. ある関数の集合に属する任意の f に対して，標準モデルでは $(2.a/b/c, f)$ が実現不可能であるが，*HSS* モデルでは $(2.a/b/c, f)$ を *semi-honest* 攻撃者に対して安全に計算できる．

関数 vecxor は，$(x_1, \cdots, x_n)$ と $(y_1, \cdots, y_m)$ を入力として受け取り，$(x_1 \oplus y_1, \cdots, x_{\min(n,m)} \oplus y_{\min(n,m)})$ を出力する関数である．標準モデルにおいて，$(2.a/c, \mathsf{vecxor})$ が実現可能で，$(2.b, \mathsf{vecxor})$ が実現不可能であることが示されている．一方 HSS モデルにおいては，2.a/b/c のいずれのクラスでも秘密計算が可能であるが，その証明は本稿では省略する．

5 実現不可能性に関する結果

この章では，HSS モデルにおける実現不可能性に関する結果を示す．5.1 節では，HSS モデルの下で $(1.d, \mathsf{omprf})$ は実現不可能であることを示す．5.2 節では，ある性質を持つ f に対して HSS モデルの下で $(2.a/b/c, f)$ が実現不可能であることを述べる．

5.1 クラス 1.d の不可能性

クラス 1.d のサイズ秘匿について計算不可能な例を示すために，関数 omprf (Oblivious Multi-point PRF-evaluation) を以下のように定義する．$\{F^\kappa\}_{\kappa \in \mathbb{N}}$ を $\{0,1\}^\kappa \times \{0,1\}^\kappa \to \{0,1\}^\kappa$ の擬似ランダム関数族とする．擬似ランダム関数族 $\{F^\kappa\}_{\kappa \in \mathbb{N}}$ に対して関数 omprf とは，2 つの入力 $x \in \{0,1\}^\kappa$ と $Y = \{y_1, \cdots, y_m\}$(ただし $y_i \in \{0,1\}^\kappa$) を受け取り，$Z = \{z_1, \cdots, z_m\}$(ただし $z_i = F^\kappa_x(y_i)$) を出力する関数である．この関数 omprf について，以下の定理が成り立つ．

定理 5.1. 一方向性関数族の存在を仮定すると，*HSS* モデルの下で $(1.d, \mathsf{omprf})$ を *semi-honest* 攻撃者に対して安全に計算することはできない．

証明. HSS モデルの下で $(1.d, \mathsf{omprf})$ を semi-honest 攻撃者に対して安全に計算するプロトコル π が存在すると仮定する (背理法の仮定)．P_1 が直接 P_2 に送るデータ量の確率変数を $T_1(\kappa, x, Y)$，P_1 が $\mathcal{F}_{\mathsf{HSS}}$ に SET によって送るデータ量の確率変数を $U_1(\kappa, x, Y)$，P_1 が $\mathcal{F}_{\mathsf{HSS}}$ から REQ によっ

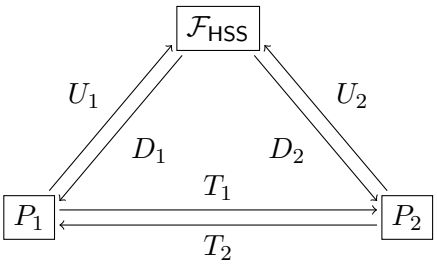

図 2: HSS モデルの通信

て受け取るデータ量の確率変数を $D_1(\kappa, x, Y)$ とする (図 2)．同様に，T_2, U_2, D_2 を定義する．Y が空集合 ϕ のとき，定義 2.1 より P_1, P_2 の実行時間は $\kappa, |x|$ の多項式でバウンドされる．$|x|$ は定数サイズであるため，ある多項式 p が存在して $T_1(\kappa, x, \phi) < p(\kappa)$ および $U_1(\kappa, x, \phi) < p(\kappa)$ が成り立つ．P_1 は $|Y|$ のサイズに関する情報を一切知り得ないので，任意の Y に対して $T_1(\kappa, x, Y) < p(\kappa)$ および $U_1(\kappa, x, Y) < p(\kappa)$ が成り立つ．もしそうでないとすると，P_1 はデータ量を比較することで，P_2 の入力が ϕ であるのか否かが判別できてしまうからである．また，一般性を失わずに $D_2(\kappa, x, Y) \leq U_1(\kappa, x, Y)$ を仮定できるので，$D_2(\kappa, x, Y) < p(\kappa)$ も成り立つ．

次に，π のシミュレータを用いて，F_x^κ の擬似乱数性を破る帰着アルゴリズム $\mathcal{D}$ を構成する．シミュレータ $\mathcal{S}_2$ は，$Y, 1^\kappa, Z = \{F_x^\kappa(y)\}_{y \in Y}$ を受け取り，T_1', T_2', U_2', D_2' および P_2 のランダムテープを出力する．シミュレーションが正しく行えているならば，$T_1'(\kappa, x, Y), D_2'(\kappa, x, Y)$ と $Y, 1^\kappa$ および P_2 のランダムテープから Z が効率的に計算できる．ここで，P_2 のランダムテープは擬似乱数でも問題ないはずであり，この場合ランダムテープは乱数のシード $s \in \{0,1\}^\kappa$ と同一視できる．したがって，ある定数 c が存在して十分大きな全ての κ に対して $T_1' + D_2' + s < \kappa^c$ となる．この S_2 を用いて擬似ランダム関数族 $\{F^\kappa\}$ と真のランダム関数族 $\{R^\kappa\}$ を識別するアルゴリズム $\mathcal{D}$ を構成する．$\mathcal{D}$ は，関数 f ($f \in F^\kappa$ か $f \in R^\kappa$) にアクセスし，$|Y| > \kappa^c$ に対して $Z = \{f(y)\}_{y \in Y}$ を受け取り，$\mathcal{S}_2$ に $Y, 1^\kappa, Z = \{f(y)\}_{y \in Y}$ を入力する．f が擬似ランダム関数のときは，$\mathcal{S}_2$ は正しくシミュレートを行う．一方，f が真のランダム関数のときは，κ^c ビットより大きい真乱数列を κ^c ビット以下に圧縮することは不可能であることから，正しくシミュレートを実行できない．よって $\mathcal{D}$ は擬似ランダム関数と真のランダム関数を識別することができる．擬似ランダム関数族の存在性と一方向性関数族の存在性は等価であるので，一方向性関数族が存在するならば，HSS モデルの下で $(1.d, \mathsf{omprf})$ を semi-honest 攻撃者に対して安全に計算するプロトコル π は存在しないことが示された．□

5.2 クラス 2 の不可能性

[LNO13] は，クラス 2 の不可能性の結果である以下の定理を示した．

定理 5.2 ([LNO13])．$f(x, y)$ を出力が固定長の関数とする．もしある正定数 $\epsilon > 0$ が存在して f の乱択通信複雑性が $\Omega(n^\epsilon)$[6] ならば，f をクラス *2.a, 2.b, 2.c* のいずれにおいても *semi-honest* 攻撃者に対して安全に計算することはできない．

HSS モデルにおいても同様の定理が成り立つ．

定理 5.3. *HSS* モデルの下でも，定理 *5.2* と同じ設定の秘密計算は不可能である．

この定理の証明は，[LNO13] の証明とほぼ同様に示すことができるため，本稿では省略する．

謝辞

価値あるコメントをいただいた新明るい暗号勉強会の皆様に深く感謝する．本研究の一部は JSPS 科研費 26330151，公益財団法人倉田記念日立科学技術財団倉田奨励金，JSPS A3 Foresight Program の助成を受けたものである．

参考文献

[ACT11] Giuseppe Ateniese, Emiliano De Cristofaro, and Gene Tsudik. (if) size matters: Size-hiding private set intersection. In *Public Key Cryptography*, pages 156–173, 2011.

[COV15] Melissa Chase, Rafail Ostrovsky, and Ivan Visconti. Executable Proofs, Input-Size Hiding Secure Computation and a New Ideal World. In *EUROCRYPT*, pages 532–560, 2015.

[IP07] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In *TCC*, pages 575–594, 2007.

[LNO13] Yehuda Lindell, Kobbi Nissim, and Claudio Orlandi. Hiding the Input-Size in Secure Two-Party Computation. In *ASIACRYPT*, pages 421–440, 2013.

[MRK03] Silvio Micali, Michael O. Rabin, and Joe Kilian. Zero-knowledge sets. In *FOCS*, pages 80–91, 2003.

[6] 関数 $f(x, y)$ の乱択通信複雑性が $\Omega(g(n))$(ただし $n = \min(|x|, |y|)$) であるとは，f を計算する任意のプロトコルについて、参加者同士が交換する情報量が $\Omega(g(n))$ ビットであることである．