マルチエージェントシステムによる電子投票プロトコルの匿名性と耐買収性の検証

川本 裕輔 † 真野 健 †† 櫻田 英樹 †† 萩谷 昌己 † ††

† 東京大学大学院情報理工学系研究科
†† NTT コミュニケーション科学基礎研究所

2007 年 3 月 4 日

概要 (1/2)

- 目標：投票プロトコルの匿名性/耐買収性検証
- 道具1：マルチエージェントシステム
- 道具2：知識論理

概要 (2/2)

- マルチエージェントシステムの枠組みで暗号関数の部分知識を扱うモデルを導入
- このモデルを能動的攻撃者の挙動を扱えるように拡張
- このモデルで, プロトコルの匿名性と耐買収性を定式化し検証する手法を提案
- [Lee et al. ’03] の耐買収性検証に適用

目次

- **検証したい性質**
- Runs and Systems
- **関数部分知識**
- **匿名性検証**
- **耐買収性検証**
- **まとめと今後の課題**

投票プロトコルに求められる性質

- 適格性
- 匿名性
- 無証拠性・耐買収性
- 公平性
- 個別検証可能性
- 総合検証可能性

検証したい性質 (1/3)

- **匿名性** (Anonymity)
  - **「誰がどの候補に投票したのか分からない」**
  - **攻撃者が選挙の様子を外から眺めても分からない**

検証したい性質 (2/3)

- **無証拠性** (Receipt-Freeness)
  - **「投票の証拠が残らない」**
  - **攻撃者が**
    - **「投票者の使ったデータ」を手に入れても，匿名性が成り立つ．**
  - **票の買収を防ぐのに必要な性質**

検証したい性質 (3/3)

- **耐買収性** (Coercion-Resistance)
  - **「買収に応じたかどうか分からない」**
  - **攻撃者が**
    - **「投票者の使ったデータ」を手に入れても，**
    - **「投票者が使うデータ」を指定しても，**

    匿名性が成り立つ．

    ([Delaune-Kremer-Ryan ’05, ’06] **による定義)**

研究の背景 (1/3)

- 匿名性 / 無証拠性 / 耐買収性の定式化と形式的検証が始まったのは**比較的最近**
  - 暗号の分野でも検証はあまり行われていないので，形式的検証に意義がある

研究の背景 (2/3)

- **プロセス代数を用いた定式化と検証が盛ん**
  - **匿名性** [Schneider-Sidiropoulos ’96]
  - **無証拠性 / 耐買収性** [Delaune-Kremer-Ryan ’05, ’06]
- **マルチエージェントシステム（MAS）と知識論理を用いた定式化の研究がある**
  - **匿名性** ([Syverson-Stubblebine ’99], [Halpern-O’Neill ’03])
  - **無証拠性** ([Jonker-Pieters ’06])

研究の背景 (3/3)

- MAS / **知識論理を用いる利点**
  - **色々な種類の知識の性質を表現**
  - **知識の性質を比較する上で便利**
- **本研究では**MAS / **知識論理を用いる**
  - Runs and Systems Framework ([Fagin et al. ’95]) **を用いる**

目次

- 検証したい性質
- Runs and Systems
- 関数部分知識
- 匿名性検証
- 耐買収性検証
- まとめと今後の課題

Runs and Systems (1/5)

- Runs and Systems Framework **の概要**
  - **各エージェントはアクションを実行する**
  - **各エージェントは局所状態を持つ**
  - **各エージェントは観測を局所状態に記録する**
  - **エージェントの知識は局所状態を用いて定義**

Runs and Systems (2/5)

- 知識の定義には可能世界の概念を利用
  - 攻撃者が区別できない可能世界を考える
    (攻撃者の局所状態の記録内容が同じ)

V_1 voted for X_1

V_2 voted for X_2

V_1 voted for X_3

V_2 voted for X_2

V_1 voted for X_2

V_2 voted for X_2

Runs and Systems (3/5)

- Runs and Systems Framework **の概要**
  - **ラン：プロトコル実行の状態を時系列に並べたもの**
  - **システム：可能なランの集合**
- Runs and Systems Framework **における匿名性の検証は「攻撃者が持つ知識」の議論**

Runs and Systems (4/5)

- Runs and Systems Framework **を用いる利点**
  - **知識論理の意味論を与える**
  - **様々な性質の定式化の研究が行われてきた**
  - **ストランド空間モデルの拡張になっており，従来技術を検証に利用できる**

Runs and Systems (5/5)

- Runs and Systems Framework **ではまだ定式化が中心で，きちんとした検証はあまり行われていない**
  - **能動的攻撃者の下での検証は行われていない**

能動的攻撃者(Dolev-Yao)の定式化

- 手持ちのデータだけを用いて，任意の通信アクションとデータ操作アクションを実行
- 公開通信路に流れるすべてのメッセージを制御
  - 攻撃者がすべてのメッセージをいったん受信
  - その後，メッセージを受信者に送ったり，第三者に渡したり，偽造したりできる

目次

- 検証したい性質
- Runs and Systems
- **関数部分知識**
- 匿名性検証
- 耐買収性検証
- まとめと今後の課題

匿名性検証で扱う知識

- 匿名性等はデータ知識の議論だけではできない
  - データ知識：局所状態に記録されたデータ
- 匿名性は「投票者と投票内容の間の関係」の分からなさ
  - データ間の関係についての知識が必要
  - **関数部分知識**を新たに導入
  - Runs and Systems Framework の中で関数部分知識を定義する

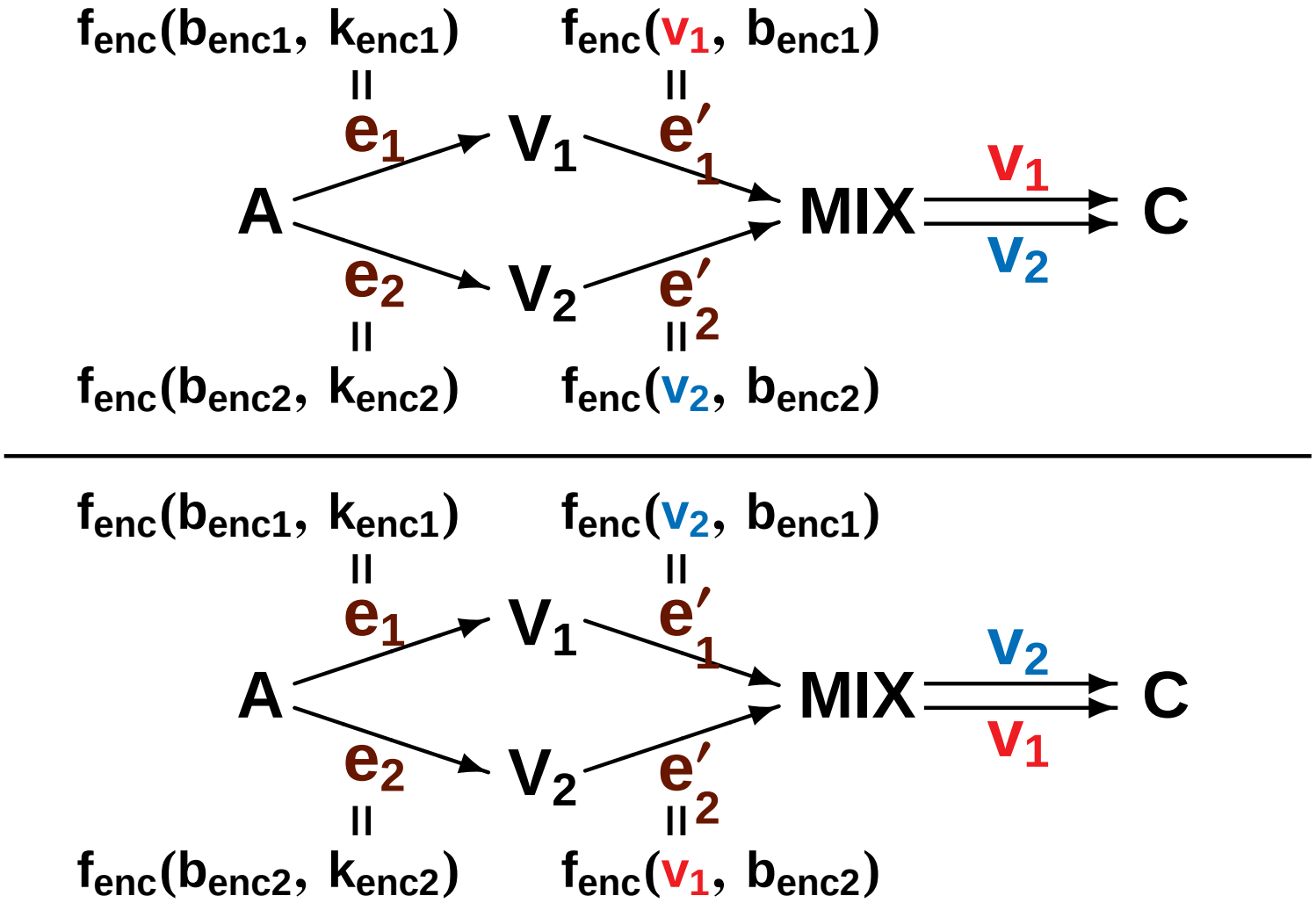
関数解釈の入れ換え (1/4)

- 投票プロトコルの匿名性を示すために
  - 攻撃者から見ると同じに見える
  - 投票者と投票内容の対応が入れ替わっている

  ような2つの投票プロトコルを用意する

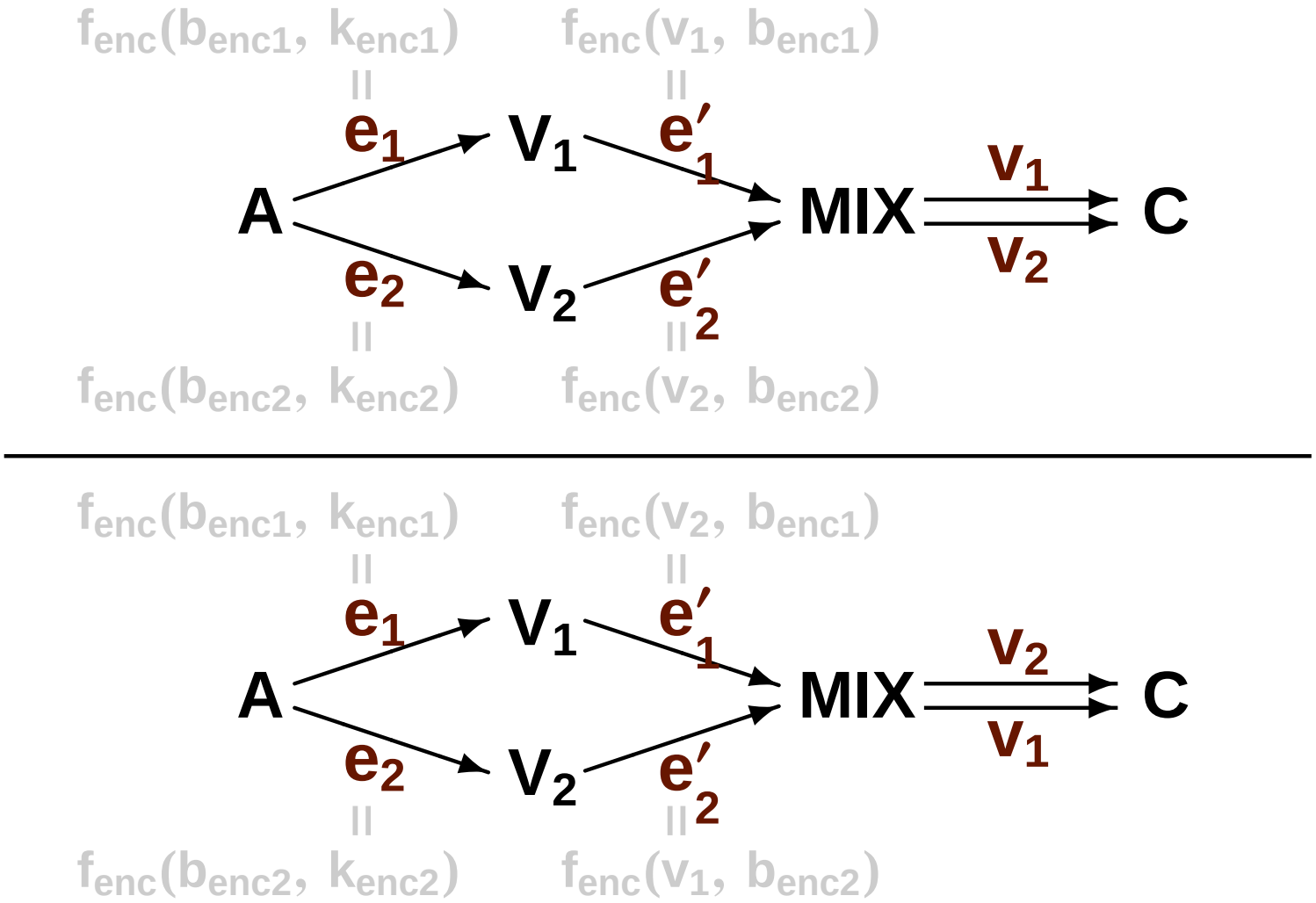
関数解釈の入れ換え (2/4)

- 投票者が 2 人 (V_1 と V_2) の選挙の例：



関数解釈の入れ換え (3/4)

- 攻撃者 I から見ると，2 つのプロトコル実行は同じに見える

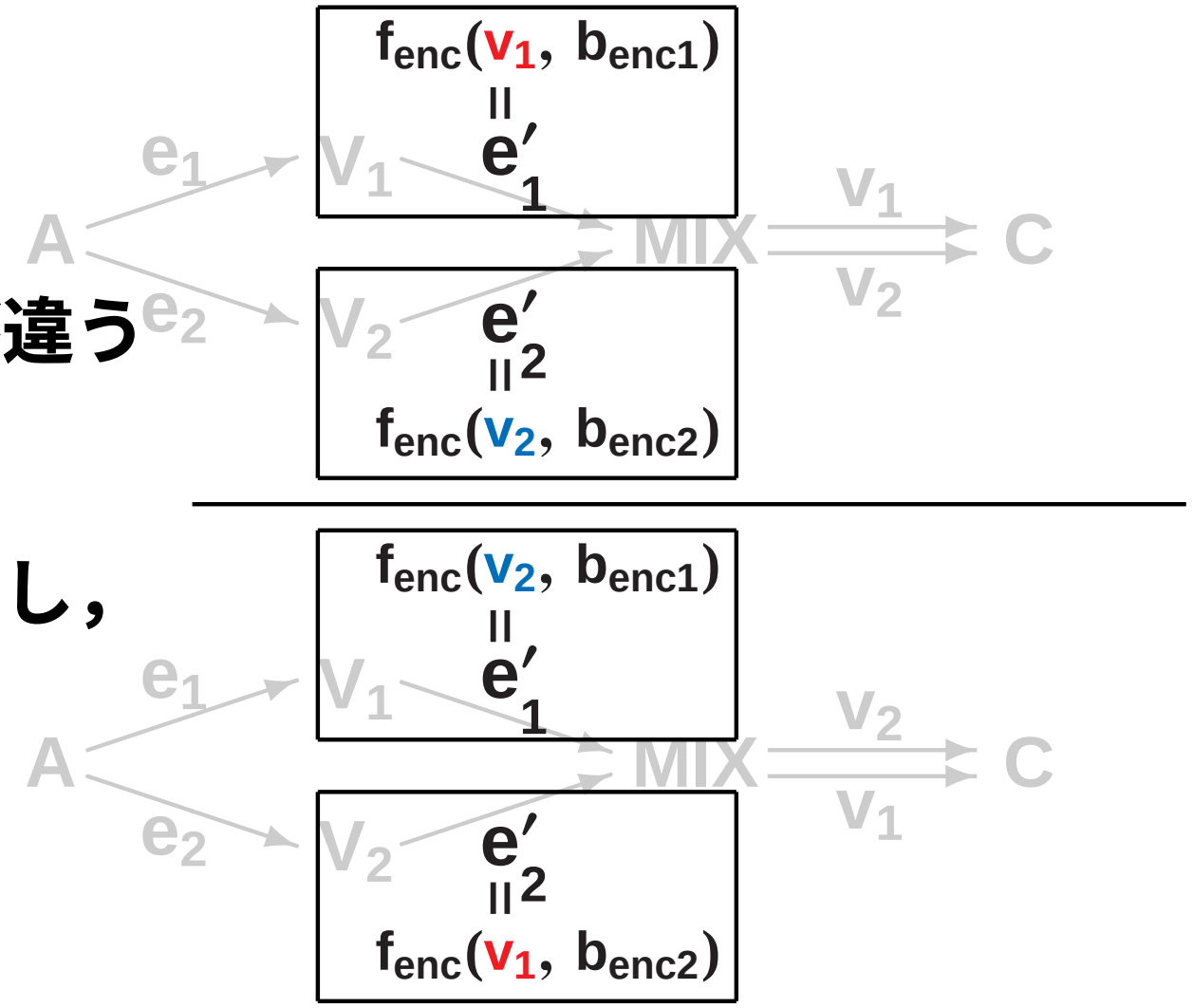


関数解釈の入れ換え (4/4)

- 攻撃者 I から見えない部分だけが違っている

同じ**暗号関数の解釈**が違う

攻撃者 I から見ると，
上の解釈かもしれないし，
下の解釈かもしれない



関数部分知識モデル (1/3)

- これまでの直観的な説明を定式化するために，Runs and Systems Framework で MAS モデルを考える
- 「暗号関数の解釈が違う可能世界」を考える
- MAS が「暗号関数のあらゆる解釈」に対応する可能世界からできているとする

関数部分知識モデル (2/3)

- 暗号アクションを実行することで
  対応する暗号関数の部分知識を手に入れる
- 暗号アクションを実際に実行するまでは
  対応する暗号関数の部分知識が手に入らない

関数部分知識モデル (3/3)

- 「関数部分知識」と「関数解釈の入れ換え」をMAS 上で厳密に議論できるモデルを作ったという点が重要
  - 準同型暗号を用いるプロトコルのように，データ間に成り立つ関係が複雑になっても，厳密な議論ができる．
- この関数部分知識モデル上で匿名性検証を行う

目次

- 検証したい性質
- Runs and Systems
- 関数部分知識
- **匿名性検証**
- 耐買収性検証
- まとめと今後の課題

匿名性検証手法 (1/4)

- 能動的攻撃者の振舞いを場合分けし，
  その場合分けごとに匿名性を示す

匿名性検証手法 (2/4)

- 投票者と投票内容の対応を入れ換えた2つのプロトコル P, P′ を用意
  - P と P′ の正規の参加者の通信アクションが攻撃者から見て同じ
- P, P′ に対応する関数解釈を定義
  - P : $f_{enc}(v_1, b_{enc1}) = e_1$ $f_{enc}(v_2, b_{enc2}) = e_2$
  - P′: $f_{enc}(v_2, b_{enc1}) = e_1$ $f_{enc}(v_1, b_{enc2}) = e_2$
- 関数解釈が入れ替わった部分の鍵 b_{enc1}, b_{enc2} を，任意の攻撃者が手に入れないことを証明
  - ストランドと同様の手法で証明

匿名性検証手法 (3/4)

- 攻撃者がP, P' のプロトコル実行を区別できると仮定し，矛盾を導く
- 最初にP と P' のプロトコル実行に食い違いが生じるのは，関数解釈が入れ替わった部分に対応する暗号アクションを実行したとき

匿名性検証手法 (4/4)

- しかし, 攻撃者は, 関数解釈が入れ替わった部分の鍵 b_{enc1}, b_{enc2} を手に入れることができないので，b_{enc1}, b_{enc2} を使った暗号アクションが実行できない
- 攻撃者は，いかなるアクションを実行しても，関数解釈の入れ替わった部分には到達できず，2つプロトコル実行が区別できない

目次

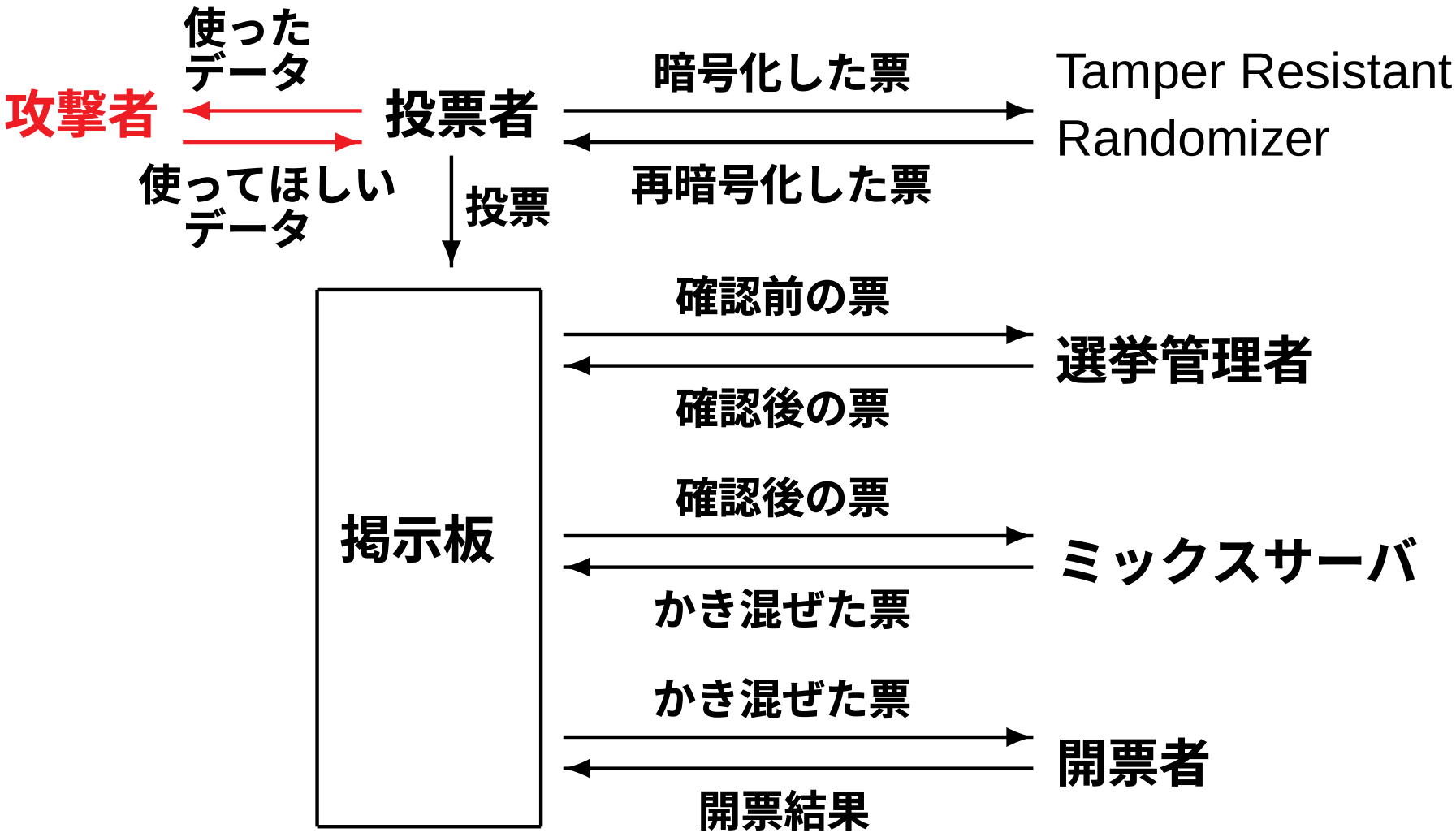
- 検証したい性質
- Runs and Systems
- 関数部分知識
- 匿名性検証
- **耐買収性検証**
- まとめと今後の課題

耐買収性検証手法 (1/3)

- 耐買収性の定式化
  - 投票プロトコルと買収プロトコルを同時に実行したときの匿名性

耐買収性検証手法 (2/3)

- [Lee et al. ’03] の耐買収性を検証した



耐買収性検証手法 (3/3)

- 買収プロトコルを実行したとしても，
  「投票者が買収に応じるプロトコル実行」と
  「投票者が買収に応じないプロトコル実行」が
  同じに見えて区別できないことを示せばよい
- 今回の検証例では耐買収性検証を匿名性検証
  に帰着できる

耐買収性の検証例 投票プロトコルを定式化

$P^{phase1}_{V_i}$:
choose($n_{\alpha i}$, V_i);
encrypt($\langle v_{X\,i}, n_{\alpha i}, k_{encC}\rangle$, e_{xyi}, V_i);
$send_{prv}(e_{xyi}, V_i, TRR_i)$;

$P^{phase2}_{TRR_i}$:
$recv_{prv}(e_{xyi}, TRR_i, V_i)$;
choose($n_{\beta i}$, TRR_i);
reencrypt($\langle e_{xyi}, n_{\beta i}, k_{encC}\rangle$, e_{xfyfi}, TRR_i);
sig($\langle e_{xfyfi}, k_{sigTRRi}\rangle$, s_{TRRi}, TRR_i);
$sig_{dvrp}(\langle e_{xyi}, e_{xfyfi}, n_{\beta i}, k_{encVi}\rangle, e_{DVRPi}, TRR_i)$;
concat($\langle e_{xfyfi}, s_{TRRi}, e_{DVRPi}\rangle$, e_{TRRi}, TRR_i);
$send_{prv}(e_{TRRi}, TRR_i, V_i)$;

$P^{phase3}_{V_i}$:
$recv_{prv}(e_{TRRi}, TRR_i, V_i)$;
split(e_{TRRi}, $\langle e_{xfyfi}, s_{TRRi}, e_{DVRPi}\rangle$, TRR_i);
ver($\langle e_{xfyfi}, s_{TRRi}, k_{verTRRi}\rangle$, c_{accept}, V_i);
$ver_{dvrp}(\langle e_{DVRPi}, e_{xyi}, e_{xfyfi}, k_{encVi}\rangle, c_{accept}, V_i)$;
sig($\langle s_{TRRi}, k_{sigVi}\rangle$, $s_{V\,TRRi}$, V_i) :
concat($\langle e_{xfyfi}, s_{TRRi}, s_{V\,TRRi}\rangle$, e_i, V_i);
$send_{prv}(e_i, V_i, BB)$;

$P^{phase4}_{BB_i}$:
$recv_{prv}(e_i, BB_i, V_i)$;
$send_{prv}(e_i, BB_i, \langle A, V_1, \cdots, V_n, I\rangle)$;

$P^{phase5}_{V_j}$:
$recv_{prv}(e_i, V_j, BB)$;

$P^{phase5}_{A_i}$:
$recv_{prv}(e_i, A, BB)$;
split(e_i, $\langle e_{xfyfi}, s_{TRRi}, s_{V\,TRRi}\rangle$, A);
ver($\langle e_{xfyfi}, s_{TRRi}, k_{verTRRi}\rangle$, c_{accept}, A);
ver($\langle s_{TRRi}, s_{V\,TRRi}, k_{verVi}\rangle$, c_{accept}, A);
$send_{prv}(e_{xfyfi}, A, BB)$;

$P^{phase6}_{BB_i}$:
$recv_{prv}(e_{xfyfi}, BB, A)$;
$send_{prv}(e_{xfyfi}, BB, \langle A, V_1, \cdots, V_n, MIX, I\rangle)$;

$P^{phase7}_{V_j}$:
$recv_{prv}(e_{xfyfi}, V_j, BB)$;

$P^{phase7}_{A_i}$:
$recv_{prv}(e_{xfyfi}, A, BB)$;

$P^{phase7}_{MIX_i}$:
$recv_{prv}(e_{xfyfi}, MIX, BB)$;
choose(n_{MIXi}, MIX);
reencrypt($\langle e_{xfyfi}, n_{MIXi}, k_{encC}\rangle$, e_{MIXi}, MIX);

$P^{phase8}_{MIX_i}$:
$send_{prv}(e_{MIXi}, MIX, BB)$;

$P^{phase9}_{BB_i}$:
$recv_{prv}(e_{MIXi}, BB, MIX)$;
$send_{prv}(e_{MIXi}, BB, \langle A, V_1, \cdots, V_n, C, I\rangle)$;

$P^{phase10}_{C_i}$:
$recv_{prv}(e_{MIXi}, C, BB)$;
decrypt($\langle e_{MIXi}, k_{decC}\rangle$, $v_{X\,i}$, C);
$check_{vote}(v_{X\,i}, C)$;
$send_{prv}(v_{X\,i}, C, BB)$;

耐買収性の検証例 買収プロトコルを定式化

$Q_{V_i}^{phase1}$:

$\mathrm{send}_{prv}(n_{\alpha i}, V_i, I)$;

$Q_{I_i}^{phase2}$:

$\mathrm{recv}_{prv}(n_{\alpha i}, I, V_i)$;

$\mathrm{choose}(\tilde{n}_{\alpha i}, I)$;

$\mathrm{concat}(\langle \tilde{n}_{\alpha i}, \tilde{v}_{X\,i} \rangle, e_{coerce}, I)$;

$\mathrm{send}_{prv}(e_{coerce}, I, V_i)$;

$Q_{V_i}^{phase3}$:

$\mathrm{recv}_{prv}(e_{coerce}, V_i, I)$;

$\mathrm{split}(e_{coerce}, \langle \tilde{n}_{\alpha i}, \tilde{v}_{X\,i} \rangle, V_i)$;

$\mathrm{encrypt}(\langle \tilde{v}_{X\,i}, \tilde{n}_{\alpha i}, k_{encC} \rangle, \tilde{e}_{xyi}, V_i)$;

$\mathrm{choose}(\hat{n}_{\gamma i}, V_i)$;

$\mathrm{sig}_{dvrp}(\langle \tilde{e}_{xyi}, e_{xfyfi}, \hat{n}_{\gamma i}, k_{encV} \rangle, \hat{e}_{DVRPi}, V_i)$;

$\mathrm{concat}(\langle e_{xfyfi}, s_{TRRi}, s_{V\,TRRi}, \hat{e}_{DVRPi} \rangle, e_{receipti}, V_i)$;

$\mathrm{send}_{prv}(e_{receipti}, V_i, I)$;

$Q_{I_i}^{phase4}$:

$\mathrm{recv}_{prv}(e_{receipti}, I, V_i)$;

$\mathrm{split}(e_{receipti}, \langle e_{xfyfi}, s_{TRRi}, s_{V\,TRRi}, \hat{e}_{DVRPi} \rangle, I)$;

$\mathrm{encrypt}(\langle \tilde{v}_{X\,i}, \tilde{n}_{\alpha i}, k_{encC} \rangle, \tilde{e}_{xyi}, I)$;

$\mathrm{ver}_{dvrp}(\langle \hat{e}_{DVRPi}, \tilde{e}_{xyi}, e_{xfyfi}, k_{decVi} \rangle, c_{accept}, I)$;

$\mathrm{concat}(\langle \tilde{e}_{xfyfi}, \tilde{s}_{TRRi}, \tilde{s}_{V\,TRRi} \rangle, \tilde{e}_i, I)$;

$\mathrm{send}_{prv}(\tilde{e}_i, I, V_i)$;

$Q_{V_i}^{phase5}$:

$\mathrm{recv}_{prv}(\tilde{e}_i, V_i, I)$;

$\mathrm{split}(\tilde{e}, \langle \tilde{e}_{xfyfi}, \tilde{s}_{TRRi}, \tilde{s}_{V\,TRRi} \rangle, V_i)$;

$\mathrm{ver}(\langle \tilde{e}_{xfyfi}, \tilde{s}_{TRRi}, k_{verTRRi} \rangle, c_{accept}, V_i)$;

$\mathrm{ver}(\langle \tilde{s}_{TRRi}, \tilde{s}_{V\,TRRi}, k_{verVi} \rangle, c_{accept}, V_i)$;

耐買収性の検証例関数解釈の入れ換え

- $f_{enc}(v_{X\,i},\ f_{mul}(n_{\alpha i},\ n_{\beta i}),\ k_{encC}) = e_{xfyfi}$

  $\downarrow \qquad \downarrow$

  $f_{enc}(\tilde{v}_{X\,i},\ f_{mul}(\tilde{n}_{\alpha i},\ n_{\beta i}),\ k_{encC}) = e_{xfyfi}$

- $f_{enc}(\tilde{v}_{X\,i},\ f_{mul}(\tilde{n}_{\alpha i},\ n_{\beta i}),\ k_{encC}) = \hat{e}_{xfyfi}$

  $\downarrow \qquad \downarrow$

  $f_{enc}(v_{X\,i},\ f_{mul}(n_{\alpha i},\ n_{\beta i}),\ k_{encC}) = \hat{e}_{xfyfi}$.

目次

- 検証したい性質
- Runs and Systems
- 関数部分知識
- 匿名性検証
- 耐買収性検証
- まとめと今後の課題

まとめ (1/2)

- Runs and Systems Framework **を拡張し，「暗号関数の部分知識」と「能動的攻撃者」を厳密に取り扱えるようにした**

まとめ (2/2)

- **能動的攻撃者の下での匿名性と耐買収性をこの枠組みで定式化した**
- **能動的攻撃者の下での匿名性と耐買収性をこの枠組みで検証する手法を提案した**
- [Lee et al. ’03] **の耐買収性を検証した**

今後の課題

- この形式的検証手法を計算論的に正当化したい
  - [Abadi-Rogaway ’01] と同様の手法でできるだろう
- マルチエージェントシステムと知識論理に確率を導入し [Halpern et al. ’02]，確率的匿名性を導く

ご清聴ありがとうございました