Decidability of a Sound Set of Inference Rules for Computational Indistinguishability

Adrien Koutsos

LSV, CNRS, ENS Paris-Saclay

June 29, 2019

Introduction

Motivation

- Security protocols are distributed programs which aim at providing some security properties.
- They are extensively used, and bugs can be very costly.
- Security protocols are often short, but the security properties are complex.

$\Rightarrow$ Need to use formal methods.

Introduction

Goal of this work

We focus on *fully automatic* proofs of *indistinguishability* properties in the *computational* model:

Introduction

Goal of this work

We focus on *fully automatic* proofs of *indistinguishability* properties in the *computational* model:

- **Computational model:** the adversary is any *probabilistic polynomial time Turing machine*. This offers strong security guarantees.

Introduction

Goal of this work

We focus on *fully automatic* proofs of *indistinguishability* properties in the *computational* model:

- **Computational model:** the adversary is any *probabilistic polynomial time Turing machine*. This offers strong security guarantees.
- **Indistinguishability properties:** e.g. strong secrecy, anonymity or unlinkability.

Introduction

Goal of this work

We focus on *fully automatic* proofs of *indistinguishability* properties in the *computational* model:

- **Computational model:** the adversary is any *probabilistic polynomial time Turing machine*. This offers strong security guarantees.
- **Indistinguishability properties:** e.g. strong secrecy, anonymity or unlinkability.
- **Fully automatic:** we want a complete decision procedure.

The Private Authentication Protocol

$$
\begin{aligned}
A' &: \; n_{A'} \xleftarrow{\$} \\
B &: \; n_B \xleftarrow{\$}
\end{aligned}
$$

$$
\begin{aligned}
1 : A' \longrightarrow B \quad &: \quad \{\langle A', n_{A'} \rangle\}_{pk(B)} \\
2 : B \longrightarrow A' \quad &: \quad \begin{cases} \{\langle n_{A'}, n_B \rangle\}_{pk(A)} & \text{if } A' = A \\ \{\langle n_B, n_B \rangle\}_{pk(A)} & \text{otherwise} \end{cases}
\end{aligned}
$$

Bana-Comon Model: Messages

Messages

We use terms to model *protocol messages*, build upon:

- **Names** $\mathcal{N}$, e.g. $\mathsf{n_A}, \mathsf{n_B}$, for random samplings.
- **Function symbols** $\mathcal{F}$, e.g.:

$$\mathsf{A}, \mathsf{B}, \langle \_ , \_ \rangle, \pi_i(\_), \{\_\}_{\_}, \mathsf{pk}(\_), \mathsf{sk}(\_), \mathsf{if\_then\_else\_}, \mathsf{eq}(\_, \_)$$

- **Variables** $\mathcal{X}$.

Bana-Comon Model: Messages

Messages

We use terms to model *protocol messages*, build upon:

- **Names** $\mathcal{N}$, e.g. $\mathsf{n_A}, \mathsf{n_B}$, for random samplings.
- **Function symbols** $\mathcal{F}$, e.g.:

$$\mathsf{A}, \mathsf{B}, \langle \_ \ , \ \_ \rangle, \pi_i(\_), \{\_\}_{\_}, \mathsf{pk}(\_), \mathsf{sk}(\_), \mathsf{if\_then\_else\_}, \mathsf{eq}(\_, \_)$$

- **Variables** $\mathcal{X}$.

Examples

$$\langle \mathsf{n_A} \ , \ \mathsf{A} \rangle \qquad \pi_1(\mathsf{n_B}) \qquad \{\langle \mathsf{A'} \ , \ \mathsf{n_{A'}} \rangle\}_{\mathsf{pk(B)}}$$

Bana-Comon Model: Messages

The Private Authentication Protocol

$$1 : \mathsf{A'} \longrightarrow \mathsf{B} \quad : \quad \{\langle \mathsf{A'}\,,\, \mathsf{n}_{\mathsf{A'}} \rangle\}_{\mathsf{pk}(\mathsf{B})}$$

$$2 : \mathsf{B} \longrightarrow \mathsf{A'} \quad : \quad \begin{cases} \{\langle \boxed{\mathsf{n}_{\mathsf{A'}}}, \mathsf{n}_{\mathsf{B}} \rangle\}_{\mathsf{pk}(\mathsf{A})} & \text{if } \boxed{\mathsf{A'}} = \mathsf{A} \\ \{\langle \mathsf{n}_{\mathsf{B}}\,,\, \mathsf{n}_{\mathsf{B}} \rangle\}_{\mathsf{pk}(\mathsf{A})} & \text{otherwise} \end{cases}$$

How do we represent the adversary's inputs?

Bana-Comon Model: Messages

The Private Authentication Protocol

$$\begin{array}{lcl} 1 : \mathsf{A'} \longrightarrow \mathsf{B} & : & \{\langle \mathsf{A'} , \mathsf{n_{A'}} \rangle\}_{\mathsf{pk(B)}} \\ 2 : \mathsf{B} \longrightarrow \mathsf{A'} & : & \begin{cases} \{\langle \boxed{\mathsf{n_{A'}}}, \mathsf{n_B} \rangle\}_{\mathsf{pk(A)}} & \text{if } \boxed{\mathsf{A'}} = \mathsf{A} \\ \{\langle \mathsf{n_B} , \mathsf{n_B} \rangle\}_{\mathsf{pk(A)}} & \text{otherwise} \end{cases} \end{array}$$

How do we represent the adversary's inputs?

- We use **adversarial functions symbols**, typically **g**.
  g takes as input the current knowledge of the adversary (**the frame**).

Bana-Comon Model: Messages

The Private Authentication Protocol

$$1 : \mathsf{A'} \longrightarrow \mathsf{B} \quad : \quad \{\langle \mathsf{A'}, \mathsf{n_{A'}} \rangle\}_{\mathsf{pk(B)}}$$

$$2 : \mathsf{B} \longrightarrow \mathsf{A'} \quad : \quad \begin{cases} \{\langle \boxed{\mathsf{n_{A'}}}, \mathsf{n_B} \rangle\}_{\mathsf{pk(A)}} & \text{if } \boxed{\mathsf{A'}} = \mathsf{A} \\ \{\langle \mathsf{n_B}, \mathsf{n_B} \rangle\}_{\mathsf{pk(A)}} & \text{otherwise} \end{cases}$$

How do we represent the adversary's inputs?

- We use **adversarial functions symbols**, typically **g**.
  g takes as input the current knowledge of the adversary (**the frame**).
- Intuitively, they can be any *probabilistic polynomial time algorithm*.
- Moreover, branching of the protocol is done using if_then_else_.

Bana-Comon Model: Messages

The Private Authentication Protocol

$$
\begin{array}{llll}
1 : \mathsf{A'} \longrightarrow \mathsf{B} & : & \{\langle \mathsf{A'}\,,\, \mathsf{n_{A'}}\rangle\}_{\mathsf{pk(B)}} & \\
2 : \mathsf{B} \longrightarrow \mathsf{A'} & : & \begin{cases} \{\langle \boxed{\mathsf{n_{A'}}}, \mathsf{n_B}\rangle\}_{\mathsf{pk(A)}} & \text{if } \boxed{\mathsf{A'}} = \mathsf{A} \\ \{\langle \mathsf{n_B}\,,\, \mathsf{n_B}\rangle\}_{\mathsf{pk(A)}} & \text{otherwise} \end{cases} &
\end{array}
$$

Term Representing the Messages in PA

$$
\begin{aligned}
t_1 &= \{\langle \mathsf{A'}\,,\, \mathsf{n_{A'}}\rangle\}_{\mathsf{pk(B)}} \\
t_2 &= \text{if} \quad \mathsf{eq}(\pi_1(\mathsf{dec}(\underline{\mathbf{g}(t_1)}, \mathsf{sk(B)})); \mathsf{A}) \\
&\quad\ \text{then } \{\langle \pi_2(\mathsf{dec}(\underline{\mathbf{g}(t_1)}, \mathsf{sk(B)})), \mathsf{n_B}\rangle\}_{\mathsf{pk(A)}} \\
&\quad\ \text{else} \quad \{\langle \mathsf{n_B}\,,\, \mathsf{n_B}\rangle\}_{\mathsf{pk(A)}}
\end{aligned}
$$

Bana-Comon Model: Protocol Execution

Protocol Execution

The execution of a protocol P is a sequence of terms using adversarial function symbols:

$$u_1^P, \ldots, u_n^P$$

where u_i^P is the i-th message sent on the network by P.

Bana-Comon Model: Protocol Execution

Protocol Execution

The execution of a protocol P is a sequence of terms using adversarial function symbols:

$$u_1^P, \ldots, u_n^P$$

where u_i^P is the i-th message sent on the network by P.

Remark

This is only possible for a bounded number of messages.

Bana-Comon Model: Security Properties

Formula

Formulas are build using:

- For every $n \in \mathbb{N}$, the predicate $\sim_n$ of arity $2n$.

Bana-Comon Model: Security Properties

Formula

Formulas are build using:

- For every $n \in \mathbb{N}$, the predicate $\sim_n$ of arity $2n$.

Examples

$$\mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n'}$$

Bana-Comon Model: Security Properties

Formula

Formulas are build using:

- For every $n \in \mathbb{N}$, the predicate $\sim_n$ of arity $2n$.

Examples

$$\mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n}'$$

Privacy of the PA protocol can be expressed by the **ground** formula:

$$t_1^{\mathsf{A}}, t_2^{\mathsf{A}} \quad \sim \quad t_1^{\mathsf{C}}, t_2^{\mathsf{C}}$$

Bana-Comon Model: Security Properties

Formula

Formulas are build using:

- For every $n \in \mathbb{N}$, the predicate $\sim_n$ of arity $2n$.
- Boolean connectives $\wedge, \vee, \neg, \rightarrow$.
- First-order quantifier $\forall$.

Examples

$$\mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n'}$$

Privacy of the PA protocol can be expressed by the **ground** formula:

$$t_1^{\mathsf{A}}, t_2^{\mathsf{A}} \quad \sim \quad t_1^{\mathsf{C}}, t_2^{\mathsf{C}}$$

Unitary Inference Rules

Unitary Inference Rules

We know that some atomic formulas are valid:

- Using α-renaming of random samplings:

$$\mathsf{n_A}, \mathsf{n_B} \sim \mathsf{n_C}, \mathsf{n_D}$$

Unitary Inference Rules

Unitary Inference Rules

We know that some atomic formulas are valid:

- Using α-renaming of random samplings:

$$\mathsf{n_A}, \mathsf{n_B} \sim \mathsf{n_C}, \mathsf{n_D}$$

- Using *cryptographic assumptions* on the security primitives, e.g. if the encryption scheme is IND-CCA_1.

Unitary Inference Rules: Cryptographic Assumptions

CCA1 Rules

$$\{m_0\}_{\mathsf{pk}} \sim \quad \{m_1\}_{\mathsf{pk}}$$

Unitary Inference Rules: Cryptographic Assumptions

CCA1 Rules

$$\{m_0\}_{\mathsf{pk}} \sim \quad \{m_1\}_{\mathsf{pk}}$$

Assuming:

- sk occurs only in decryption position in m_0, m_1

Unitary Inference Rules: Cryptographic Assumptions

CCA1 Rules

$$\{m_0\}_{\mathsf{pk}}^{\mathsf{n}_r} \sim \quad \{m_1\}_{\mathsf{pk}}^{\mathsf{n}_r}$$

Assuming:

- sk occurs only in decryption position in m_0, m_1
- n_r does not appear in m_0, m_1

Unitary Inference Rules: Cryptographic Assumptions

CCA1 Rules

$$\{m_0\}_{\mathsf{pk}}^{\mathsf{n}_r} \sim \quad \{m_1\}_{\mathsf{pk}}^{\mathsf{n}_r}$$

Assuming:

- sk occurs only in decryption position in m_0, m_1
- n_r does not appear in m_0, m_1

Theorem

The CCA1 rules are valid when the encryption and decryption functions form an IND-CCA_1 encryption scheme.

Unitary Inference Rules: Cryptographic Assumptions

CCA1 Rules

$$\vec{v},\ \{m_0\}_{\mathsf{pk}}^{\mathsf{n}_r} \sim \vec{v},\ \{m_1\}_{\mathsf{pk}}^{\mathsf{n}_r}$$

Assuming:

- sk occurs only in decryption position in $m_0, m_1, \vec{v}$
- n_r does not appear in $m_0, m_1, \vec{v}$

Theorem

The CCA1 rules are valid when the encryption and decryption functions form an $\textsc{ind-cca}_1$ encryption scheme.

Unitary Inference Rules: Cryptographic Assumptions

CCA1 Rules

$$\vec{v},\ \{m_0\}_{\mathsf{pk}}^{\mathsf{n}_r} \sim \vec{v},\ \{m_1\}_{\mathsf{pk}}^{\mathsf{n}_r}$$

Assuming:

- sk occurs only in decryption position in $m_0, m_1, \vec{v}$
- n_r does not appear in $m_0, m_1, \vec{v}$

Theorem

The CCA1 rules are valid when the encryption and decryption functions form an IND-CCA_1 encryption scheme.

Remark

This is an axiom schema!

Inference Rules

Proof Technique

- If $\vec{u} \sim \vec{v}$ is not directly valid, we try to prove it through a succession of *rule applications*:

$$\frac{\vec{s} \sim \vec{t}}{\vec{u} \sim \vec{v}}$$

- This is the way cryptographers do proofs.

Inference Rules

Proof Technique

- If $\vec{u} \sim \vec{v}$ is not directly valid, we try to prove it through a succession of *rule applications*:

$$\frac{\vec{s} \sim \vec{t}}{\vec{u} \sim \vec{v}}$$

- This is the way cryptographers do proofs.
- **Validity by reduction:** given a winning adversary against $\vec{u} \sim \vec{v}$, we can build winning adversary againstan adversary winning $\vec{s} \sim \vec{t}$.

Inference Rules

Proof Technique

- If $\vec{u} \sim \vec{v}$ is not directly valid, we try to prove it through a succession of *rule applications*:

$$\frac{\vec{s} \sim \vec{t}}{\vec{u} \sim \vec{v}}$$

- This is the way cryptographers do proofs.
- **Validity by reduction:** given a winning adversary against $\vec{u} \sim \vec{v}$, we can build winning adversary againstan adversary winning $\vec{s} \sim \vec{t}$.

Example

$$\frac{x \sim y}{y \sim x} \; Sym$$

Structural Rules

Duplicate

$$\frac{x \sim \quad y}{x, x \sim \quad y, y}\ \text{Dup}$$

Structural Rules

Duplicate

$$\frac{\vec{w}_l, x \sim \vec{w}_r, y}{\vec{w}_l, x, x \sim \vec{w}_r, y, y}\ \mathsf{Dup}$$

Structural Rules

Function Application

If you cannot distinguish the arguments, you cannot distinguish the images.

$$\frac{x_1, \ldots, x_n \sim y_1, \ldots, y_n}{f(x_1, \ldots, x_n) \sim f(y_1, \ldots, y_n)}\ \mathsf{FA}$$

Structural Rules

Function Application

If you cannot distinguish the arguments, you cannot distinguish the images.

$$\frac{\vec{w}_l, x_1, \ldots, x_n \sim \vec{w}_r, y_1, \ldots, y_n}{\vec{w}_l, f(x_1, \ldots, x_n) \sim \vec{w}_r, f(y_1, \ldots, y_n)} \text{ FA}$$

Structural Rules

Case Study

If we use Function Application on if_then_else_:

$$\frac{b, u, v \sim b', u', v'}{\text{if } b \text{ then } u \text{ else } v \sim \text{if } b' \text{ then } u' \text{ else } v'} \text{ FA}$$

Structural Rules

Case Study

If we use Function Application on if_then_else_:

$$\frac{b, u, v \sim b', u', v'}{\text{if } b \text{ then } u \text{ else } v \sim \text{if } b' \text{ then } u' \text{ else } v'}\ \text{FA}$$

But we can do better:

$$\frac{b, u \sim b', u' \qquad\qquad b, v \sim b', v'}{\text{if } b \text{ then } u \text{ else } v \sim \text{if } b' \text{ then } u' \text{ else } v'}\ \text{CS}$$

Rewriting Rules

Remark: $\sim$ is not a congruence!

Counter-Example: $\mathsf{n} \sim \mathsf{n}$ and $\mathsf{n} \sim \mathsf{n}'$, but $\mathsf{n}, \mathsf{n} \not\sim \mathsf{n}, \mathsf{n}'$.

Rewriting Rules

Remark: $\sim$ is not a congruence!

Counter-Example: $\mathsf{n} \sim \mathsf{n}$ and $\mathsf{n} \sim \mathsf{n}'$, but $\mathsf{n}, \mathsf{n} \not\sim \mathsf{n}, \mathsf{n}'$.

Congruence

If $\mathrm{eq}(u; v) \sim \mathsf{true}$ then u and v are (almost always) *equal*
$\Rightarrow$ we have a congruence.

$u = v$ syntactic sugar for $\mathrm{eq}(u; v) \sim \mathsf{true}$

Equational Theory: Protocol Functions

- $\pi_i\left(\langle x_1, x_2\rangle\right) = x_i$ $\qquad i \in \{1, 2\}$
- $\mathrm{dec}(\{x\}_{\mathrm{pk}(\mathsf{y})}, \mathrm{sk}(\mathsf{y})) = x$

Rewriting Rules

Equational Theory: Protocol Functions

If Homomorphism:

$$f(\vec{u}, \text{if } b \text{ then } x \text{ else } y, \vec{v}) = \text{if } b \text{ then } f(\vec{u}, x, \vec{v}) \text{ else } f(\vec{u}, y, \vec{v})$$

$$\text{if } (\text{if } b \text{ then } a \text{ else } c) \text{ then } x \text{ else } y = \\ \text{if } b \text{ then } (\text{if } a \text{ then } x \text{ else } y) \text{ else } (\text{if } c \text{ then } x \text{ else } y)$$

Rewriting Rules

Equational Theory: Protocol Functions

If Homomorphism:

$f(\vec{u}, \text{if } b \text{ then } x \text{ else } y, \vec{v}) = \text{if } b \text{ then } f(\vec{u}, x, \vec{v}) \text{ else } f(\vec{u}, y, \vec{v})$

$\text{if } (\text{if } b \text{ then } a \text{ else } c) \text{ then } x \text{ else } y =$

$\qquad \text{if } b \text{ then } (\text{if } a \text{ then } x \text{ else } y) \text{ else } (\text{if } c \text{ then } x \text{ else } y)$

If Rewriting:

$\text{if } b \text{ then } x \text{ else } x = x$

$\text{if } b \text{ then } (\text{if } b \text{ then } x \text{ else } y) \text{ else } z = \text{if } b \text{ then } x \text{ else } z$

$\text{if } b \text{ then } x \text{ else } (\text{if } b \text{ then } y \text{ else } z) = \text{if } b \text{ then } x \text{ else } z$

Rewriting Rules

Equational Theory: Protocol Functions

If Homomorphism:

$$f(\vec{u}, \text{if } b \text{ then } x \text{ else } y, \vec{v}) = \text{if } b \text{ then } f(\vec{u}, x, \vec{v}) \text{ else } f(\vec{u}, y, \vec{v})$$

$$\text{if } (\text{if } b \text{ then } a \text{ else } c) \text{ then } x \text{ else } y = \\ \text{if } b \text{ then } (\text{if } a \text{ then } x \text{ else } y) \text{ else } (\text{if } c \text{ then } x \text{ else } y)$$

If Rewriting:

$$\text{if } b \text{ then } x \text{ else } x = x$$

$$\text{if } b \text{ then } (\text{if } b \text{ then } x \text{ else } y) \text{ else } z = \text{if } b \text{ then } x \text{ else } z$$

$$\text{if } b \text{ then } x \text{ else } (\text{if } b \text{ then } y \text{ else } z) = \text{if } b \text{ then } x \text{ else } z$$

If Re-Ordering:

$$\text{if } b \text{ then } (\text{if } a \text{ then } x \text{ else } y) \text{ else } z = \\ \text{if } a \text{ then } (\text{if } b \text{ then } x \text{ else } z) \text{ else } (\text{if } b \text{ then } y \text{ else } z)$$

$$\text{if } b \text{ then } x \text{ else } (\text{if } a \text{ then } y \text{ else } z) = \\ \text{if } a \text{ then } (\text{if } b \text{ then } x \text{ else } y) \text{ else } (\text{if } b \text{ then } x \text{ else } z)$$

Decidability

Decision Problem: Unsatisfiability

Input: A ground formula $\vec{u} \sim \vec{v}$.
Question: Is there a derivation of $\vec{u} \sim \vec{v}$ using Ax?

Decidability

Decision Problem: Unsatisfiability

Input: A ground formula $\vec{u} \sim \vec{v}$.
Question: Is there a derivation of $\vec{u} \sim \vec{v}$ using Ax?

or equivalently

Decision Problem: Game Transformations

Input: A game $\vec{u} \sim \vec{v}$.
Question: Is there a sequence of game transformations in Ax showing that $\vec{u} \sim \vec{v}$ is secure?

Inference Rules: Summary

The Inference Rules in Ax

$$\frac{x \sim y}{x, x \sim y, y}\ \text{Dup}$$

$$\frac{x_1, \ldots, x_n \sim y_1, \ldots, y_n}{f(x_1, \ldots, x_n) \sim f(y_1, \ldots, y_n)}\ \text{FA}$$

$$\frac{b, u \sim b', u' \qquad b, v \sim b', v'}{\text{if } b \text{ then } u \text{ else } v \sim \text{if } b' \text{ then } u' \text{ else } v'}\ \text{CS}$$

$$\frac{\vec{u}' \sim \vec{v}'}{\vec{u} \sim \vec{v}}\ R \quad \text{when } \vec{u} =_R \vec{u}' \text{ and } \vec{v} =_R \vec{v}'$$

$$\frac{}{\vec{u} \sim \vec{v}}\ \text{CCA1}$$

Term Rewriting System

Theorem

There exists a term rewriting system $\rightarrow_R \subseteq =$ such that:

- $\rightarrow_R$ is convergent.
- $=$ is equal to $({}_R\!\leftarrow \cup \rightarrow_R)^*$.

Strategy

Deconstructing Rules

Rules CS, FA and Dup are decreasing transformations.

Strategy

Deconstructing Rules

Rules CS, FA and Dup are decreasing transformations.

Problems

- The rule R is not decreasing!
- CCA1 is a recursive schema.

Strategy

Deconstructing Rules

Rules CS, FA and Dup are decreasing transformations.

Problems

- The rule R is not decreasing!
- CCA1 is a recursive schema.

Naive Idea

R is convergent, so could we restrict proofs to terms in R-normal form?

Difficulties

If Introduction: $x \rightarrow \text{if } b \text{ then } x \text{ else } x$

$$\frac{}{\mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n'}}$$

Difficulties

If Introduction: $x \to \text{if } b \text{ then } x \text{ else } x$

$$\frac{\overline{\text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n}'}}{\mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n}'}\ R$$

Difficulties

If Introduction: $x \to \text{if } b \text{ then } x \text{ else } x$

$$\dfrac{\dfrac{\dfrac{}{\mathsf{n} \sim \mathsf{n}}}{g(), \mathsf{n} \sim g(), \mathsf{n}}\ \text{FA} \qquad \dfrac{\dfrac{}{\mathsf{n} \sim \mathsf{n}'}}{g(), \mathsf{n} \sim g(), \mathsf{n}'}\ \text{FA}}{\dfrac{\text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n}'}{\mathsf{n} \sim \text{if } g() \text{ then } \mathsf{n} \text{ else } \mathsf{n}'}\ R}\ \text{CS}$$

Difficulties

If Introduction: : $x \rightarrow$ if b then x else x

$$
\dfrac{
  \dfrac{
    \dfrac{\dfrac{}{\vec{u}, \mathsf{n} \sim \vec{u}, \mathsf{n}}}{\vec{u}, g(\vec{u}), \mathsf{n} \sim \vec{u}, g(\vec{u}), \mathsf{n}}\ \text{FA, Dup}
    \qquad
    \dfrac{\dfrac{}{\vec{u}, \mathsf{n} \sim \vec{u}, \mathsf{n}'}}{\vec{u}, g(\vec{u}), \mathsf{n} \sim \vec{u}, g(\vec{u}), \mathsf{n}'}\ \text{FA, Dup}
  }{
    \vec{u}, \text{if } g(\vec{u}) \text{ then } \mathsf{n} \text{ else } \mathsf{n} \sim \vec{u}, \text{if } g(\vec{u}) \text{ then } \mathsf{n} \text{ else } \mathsf{n}'
  }\ \text{CS}
}{
  \vec{u}, \mathsf{n} \sim \vec{u}, \text{if } g(\vec{u}) \text{ then } \mathsf{n} \text{ else } \mathsf{n}'
}\ R
$$

Difficulties

If Introduction: : $x \rightarrow$ if b then x else x

$$
\dfrac{\dfrac{\dfrac{}{\vec{u}, \mathsf{n} \sim \vec{u}, \mathsf{n}}}{\vec{u}, g(\vec{u}), \mathsf{n} \sim \vec{u}, g(\vec{u}), \mathsf{n}}\ \text{FA, Dup} \qquad \dfrac{\dfrac{}{\vec{u}, \mathsf{n} \sim \vec{u}, \mathsf{n'}}}{\vec{u}, g(\vec{u}), \mathsf{n} \sim \vec{u}, g(\vec{u}), \mathsf{n'}}\ \text{FA, Dup}}{\dfrac{\vec{u}, \text{if } g(\vec{u}) \text{ then } \mathsf{n} \text{ else } \mathsf{n} \sim \vec{u}, \text{if } g(\vec{u}) \text{ then } \mathsf{n} \text{ else } \mathsf{n'}}{\vec{u}, \mathsf{n} \sim \vec{u}, \text{if } g(\vec{u}) \text{ then } \mathsf{n} \text{ else } \mathsf{n'}}\ R}\ \text{CS}
$$

Bounded Introduction

Still, the introduced conditional $g(\vec{u})$ is bounded by the other side.

Decision Procedure

Proof Cut: Introduction of a Conditional on Both Sides

$$\dfrac{\dfrac{a, s \sim b, t \qquad\qquad a, s \sim b, t}{\text{if } a \text{ then } s \text{ else } s \sim \text{if } b \text{ then } t \text{ else } t}\ \text{CS}}{s \sim t}\ R$$

Decision Procedure

Proof Cut: Introduction of a Conditional on Both Sides

$$\dfrac{\dfrac{a, s \sim b, t \qquad\qquad a, s \sim b, t}{\text{if } a \text{ then } s \text{ else } s \sim \text{if } b \text{ then } t \text{ else } t}\ \text{CS}}{s \sim t}\ R$$

Lemma

From a proof of $a, s \sim b, t$ we can extract a smaller proof of $s \sim t$.

Decision Procedure

Proof Cut: Introduction of a Conditional on Both Sides

$$\dfrac{\dfrac{a, s \sim b, t \qquad\qquad a, s \sim b, t}{\text{if } a \text{ then } s \text{ else } s \sim \text{if } b \text{ then } t \text{ else } t}\ \text{CS}}{s \sim t}\ R$$

Lemma

From a proof of $a, s \sim b, t$ we can extract a smaller proof of $s \sim t$.

$\Rightarrow$ **Proof Cut Elimination**

Decision Procedure

Proof Cut

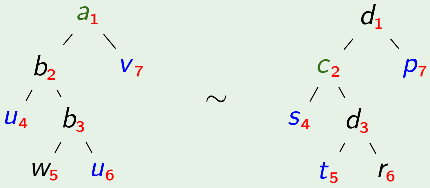
$$
\dfrac{\dfrac{a_1, b_2, b_3, u_4, w_5, u_6, v_7 \sim d_1, c_2, d_3, s_4, t_5, r_6, p_7}{
\begin{array}{ccccccc}
 & & & a_1 & & & \\
 & & / & & \backslash & & \\
 & b_2 & & & & v_7 & \\
 / & & \backslash & & & & \\
u_4 & & & b_3 & & & \\
 & & / & & \backslash & & \\
 & w_5 & & & & u_6 &
\end{array}
\quad \sim \quad
\begin{array}{ccccccc}
 & & & d_1 & & & \\
 & & / & & \backslash & & \\
 & c_2 & & & & p_7 & \\
 / & & \backslash & & & & \\
s_4 & & & d_3 & & & \\
 & & / & & \backslash & & \\
 & t_5 & & & & r_6 &
\end{array}
}\ \mathsf{FA}^{(3)}}{\text{if } a \text{ then } u \text{ else } v \sim \text{if } c \text{ then } s \text{ else } t}\ R
$$

where $p \equiv$ if c then s else t

Decision Procedure

Proof Cut

$$\dfrac{\dfrac{a_1, b_2, b_3, u_4, w_5, u_6, v_7 \sim d_1, c_2, d_3, s_4, t_5, r_6, p_7}{}\ \mathsf{FA}^{(3)}}{\text{if } a \text{ then } u \text{ else } v \sim \text{if } c \text{ then } s \text{ else } t}\ R$$



where $p \equiv \text{if } c \text{ then } s \text{ else } t$

Key Lemma

If $b, b \sim b', b''$ can be shown using only FA, Dup and CCA1 then $b' \equiv b''$.

Decision Procedure

Proof Cut

$$\dfrac{\dfrac{a_1, b_2, b_3, u_4, w_5, u_6, v_7 \sim d_1, c_2, d_3, s_4, t_5, r_6, p_7}{\begin{array}{ccccc} & & a_1 & & \\ & / & & \backslash & \\ & b_2 & & & v_7 \\ / & & \backslash & & \\ u_4 & & b_3 & & \\ & / & & \backslash & \\ & w_5 & & u_6 & \end{array} \quad \sim \quad \begin{array}{ccccc} & & d_1 & & \\ & / & & \backslash & \\ & c_2 & & & p_7 \\ / & & \backslash & & \\ s_4 & & d_3 & & \\ & / & & \backslash & \\ & t_5 & & r_6 & \end{array}}\ \mathrm{FA}^{(3)}}{\text{if } a \text{ then } u \text{ else } v \sim \text{if } c \text{ then } s \text{ else } t}\ R$$

where $p \equiv$ if c then s else t

Proof Cut Elimination

- $b_2, b_3 \sim c_2, d_3 \qquad \Rightarrow \qquad c \equiv d.$

Decision Procedure

Proof Cut

$$
\cfrac{\cfrac{a_1, b_2, b_3, u_4, w_5, u_6, v_7 \sim d_1, c_2, d_3, s_4, t_5, r_6, p_7}{
\begin{array}{ccccc}
 & & a_1 & & \\
 & / & & \backslash & \\
 & b_2 & & & v_7 \\
 / & & \backslash & & \\
u_4 & & b_3 & & \\
 & / & & \backslash & \\
 & w_5 & & u_6 &
\end{array}
\quad \sim \quad
\begin{array}{ccccc}
 & & d_1 & & \\
 & / & & \backslash & \\
 & c_2 & & & p_7 \\
 / & & \backslash & & \\
s_4 & & d_3 & & \\
 & / & & \backslash & \\
 & t_5 & & r_6 &
\end{array}
}\ \mathsf{FA}^{(3)}}{\text{if } a \text{ then } u \text{ else } v \sim \text{if } c \text{ then } s \text{ else } t}\ R
$$

where $p \equiv \text{if } c \text{ then } s \text{ else } t$

Proof Cut Elimination

- $b_2, b_3 \sim c_2, d_3 \quad \Rightarrow \quad c \equiv d.$
- $a_1, b_2 \sim d_1, c_2 \quad \Rightarrow \quad a \equiv b.$

Strategy: Theorem

Theorem

The following problem is decidable:
Input: A ground formula $\vec{u} \sim \vec{v}$.
Question: Is there a derivation of $\vec{u} \sim \vec{v}$ using Ax?

Strategy: Theorem

Theorem

The following problem is decidable:
Input: A ground formula $\vec{u} \sim \vec{v}$.
Question: Is there a derivation of $\vec{u} \sim \vec{v}$ using Ax?

Remark: Unitary Inference Rules

This holds when using CCA2 as unitary inference rules.

Strategy: Theorem

Theorem

The following problem is decidable:
Input: A ground formula $\vec{u} \sim \vec{v}$.
Question: Is there a derivation of $\vec{u} \sim \vec{v}$ using Ax?

Remark: Unitary Inference Rules

This holds when using CCA2 as unitary inference rules.

Sketch

- Commute rule applications to order them as follows:

$$(2\mathrm{Box} + R_\Box) \cdot \mathrm{CS}_\Box \cdot \mathrm{FA}_{\mathrm{if}} \cdot \mathrm{FA}_{\mathrm{f}} \cdot \mathrm{Dup} \cdot \mathrm{U}$$

- We do proof cut eliminations to get a small proof.

Conclusion

Contribution

Decidability of a set of inference rules for computational indistinguishability.

Conclusion

Contribution

Decidability of a set of inference rules for computational indistinguishability.

Limitations

- The complexity is high: 3-NEXPTIME.
- The cryptographic primitives are fixed: only for CCA2.

Conclusion

Contribution

Decidability of a set of inference rules for computational indistinguishability.

Limitations

- The complexity is high: 3-NEXPTIME.
- The cryptographic primitives are fixed: only for CCA2.

Future Works

Study the scope of the result:

- Support for a larger class of primitives and associated assumptions.
- Undecidability results for extensions of the set of axioms.

Thanks for your attention

Commutations

$(R \mid \mathrm{Dup})$ Commutation

This application

$$\dfrac{\dfrac{\vec{u}, s \sim \vec{u}', s'}{\vec{u}, t \sim \vec{u}', t'}\ R}{\vec{u}, t, t \sim \vec{u}', t', t'}\ \mathrm{Dup}$$

Commutations

$(R \mid \text{Dup})$ Commutation

This application

$$\dfrac{\dfrac{\vec{u}, s \sim \vec{u}', s'}{\vec{u}, t \sim \vec{u}', t'}\ R}{\vec{u}, t, t \sim \vec{u}', t', t'}\ \text{Dup}$$

Can be rewritten into:

$$\dfrac{\dfrac{\vec{u}, s \sim \vec{u}', s'}{\vec{u}, s, s \sim \vec{u}', s', s'}\ \text{Dup}}{\vec{u}, t, t \sim \vec{u}', t', t'}\ R$$

Commutations

$(R \mid \mathsf{FA})$ Commutation

This application:

$$\dfrac{\dfrac{\vec{u}_1, \vec{v}_1 \sim \vec{u}'_1, \vec{v}'_1}{\vec{u}, \vec{v} \sim \vec{u}', \vec{v}'}\ R}{\vec{u}, f(\vec{v}), \vec{u}', f(\vec{v}')}\ \mathsf{FA}$$

Commutations

$(R \mid \mathsf{FA})$ Commutation

This application:

$$\dfrac{\dfrac{\vec{u}_1, \vec{v}_1 \sim \vec{u}'_1, \vec{v}'_1}{\vec{u}, \vec{v} \sim \vec{u}', \vec{v}'}\ R}{\vec{u}, f(\vec{v}), \vec{u}', f(\vec{v}')}\ \mathsf{FA}$$

Can be rewritten into:

$$\dfrac{\dfrac{\vec{u}_1, \vec{v}_1 \sim \vec{u}'_1, \vec{v}'_1}{\vec{u}_1, f(\vec{v}_1) \sim \vec{u}'_1, f(\vec{v}'_1)}\ \mathsf{FA}}{\vec{u}, f(\vec{v}), \vec{u}', f(\vec{v}')}\ R$$