A decision procedure for proving symbolic equivalence

V. Cheval, H. Comon-Lundh, S. Delaune

LSV, Project SECSI

19 June 2010
Context

Automatic procedure for proving security properties on protocol

Formalism
The rules
Termination, Completeness and Soundness

Termination, Completeness and Soundness

Termination, Completeness and Soundness

V. Cheval, H. Comon-Lundh, S. Delaune

Symbolic equivalence of constraint systems
Formalism

The rules

Termination, Completeness and Soundness

Context

Automatic procedure for proving security properties on protocol

Trace properties

- Examples: simple secret, authentication, ...
- All traces of a protocol has to satisfy a certain property.
- Lot of previous works on those security properties.
- Tools already exists (example: ProVerif, Maude-NPA,...)
Formalism

Termination, Completeness and Soundness

Context

Automatic procedure for proving security properties on protocol

Trace properties
- Examples: simple secret, authentication, ...
- All traces of a protocol have to satisfy a certain property.
- Lot of previous works on those security properties.
- Tools already exist (example: ProVerif, Maude-NPA, ...)

Equivalence properties
- Examples: strong secret, dictionary attacks, anonymity, ...
- Express the indistinguishability of two protocols
- Theoretical results (Baudet, Chevalier, Rusinowitch, ...)
- No general tool implemented
Security properties example: Anonymity

0. \[ A \rightarrow B : \text{aenc}(\langle N_a, p(A) \rangle, p(B)) \]
1. \[ B \rightarrow A : \text{aenc}(\langle N_a, \langle N_b, p(B) \rangle \rangle, p(A)) \]
Security properties example: Anonymity

0. $A \rightarrow B : \text{aenc}(\langle N_a, p(A) \rangle, p(B))$
1. $B \rightarrow A : \text{aenc}(\langle N_a, \langle N_b, p(B) \rangle \rangle, p(A))$

Security property: Anonymity

The identity of the principal $A$ cannot be revealed to the attacker.
Security properties example: Anonymity

0. \( A \rightarrow B : \text{aenc}\left(\langle N_a, p(A)\rangle, p(B)\right) \)

1. \( B \rightarrow A : \text{aenc}\left(\langle N_a, \langle N_b, p(B)\rangle\rangle, p(A)\right) \)

Security property: Anonymity

The identity of the principal \( A \) cannot be revealed to the attacker.

Formally

\[
\begin{align*}
    c(p(a)).c(p(a')).c(p(b)) & \parallel P_A(a, b) \parallel P_B(b, a) \\
    \approx & \\
    c(p(a)).c(p(a')).c(p(b)) & \parallel P_A(a', b) \parallel P_B(b, a')
\end{align*}
\]
Previous works

**Huttel (2002)**
- Only spi-calculus (fixed primitives)
- Untractable implementation (multi-exponential complexity)
- Doesn’t handle trace properties.

**Blanchet, Abadi, Fournet (2008) : ProVerif**
- Unbounded number of sessions
- Diff-equivalence: Observational equivalence between two process with the same structure but different messages.
- Very efficient
- Possibility of false attacks. Doesn’t always terminate
Previous works


- Bounded number of sessions
- Infinitely many traces are represented by constraint systems
- Observational equivalence of processes ⇔ symbolic equivalence of constraint systems
- Algorithm for the symbolic equivalence of positive constraint systems when the equational theory is given by a subterm convergent rewriting system.
Outline

1. Formalism
   - Constraint Systems
   - Equivalence

2. The rules
   - Definition and example
   - How to use them?

3. Termination, Completeness and Soundness
Dolev-Yao

Rewrite rules

- \( \text{dec}(\text{enc}(x, y), y) \rightarrow x \)
- \( \text{adec}(\text{aenc}(x, p(y)), y) \rightarrow x \)
- \( \text{check}(\text{sign}(x, y), p(y)) \rightarrow x \)
- \( \pi_1(\langle x, y \rangle) \rightarrow x \) and \( \pi_2(\langle x, y \rangle) \rightarrow y \)
Constraint system

0. \( A \rightarrow B : \text{aenc}(\langle Na, p(A) \rangle, p(B)) \)

1. \( B \rightarrow A : \text{aenc}(\langle Na, Nb, p(B) \rangle, p(A)) \)

Constraint system

\[
p(A), p(B), \{\langle Na, p(A) \rangle \}_p(B) \vdash \{\langle x, y \rangle \}_p(B) \\
p(B), p(B), \{\langle Na, p(A) \rangle \}_p(B), \{\langle x, Nb, p(B) \rangle \}_y \vdash \{\langle Na, z, p(B) \rangle \}_p(A)
\]
Solution of a constraint system

\[
p(A), p(B), \{\langle N_a, p(A) \rangle \}_p(B) \vdash \{\langle x, y \rangle \}_p(B)
p(A), p(B), \{\langle N_a, p(A) \rangle \}_p(B), \{\langle x, N_b, p(B) \rangle \}_y \vdash \{\langle N_a, z, p(B) \rangle \}_p(A)
\]
Solution of a constraint system

A solution

\[ \sigma = \{ x \mapsto N_a ; \ y \mapsto p(a) ; \ z \mapsto N_b \} , \text{ and} \]

\[ \theta = \{ X_1 \mapsto ax_3 ; \ X_2 \mapsto ax_4 \} \]
Outline

1 Formalism
   - Constraint Systems
   - Equivalence

2 The rules
   - Definition and example
   - How to use them?

3 Termination, Completeness and Soundness
Static equivalence

**Static equivalence :** $\phi \sim \phi'$

Given two sequences of terms $\phi, \phi'$, the intruder cannot distinguish them.

- $\forall (\xi, \xi') \in \Pi^2, \xi \phi \downarrow = \xi' \phi \downarrow \iff \xi \phi' \downarrow = \xi' \phi' \downarrow$
- $\forall \xi \in \Pi, \xi \phi \downarrow$ is a message $\iff \xi \phi' \downarrow$ is a message
Static equivalence:

Static equivalence: \( \phi \sim \phi' \)

Given two sequences of terms \( \phi, \phi' \), the intruder cannot distinguish them.

\[
\forall (\xi, \xi') \in \Pi^2, \xi \phi \downarrow = \xi' \phi' \downarrow \iff \xi \phi' \downarrow = \xi' \phi' \downarrow
\]

\[\forall \xi \in \Pi, \xi \phi \downarrow \text{ is a message} \iff \xi \phi' \downarrow \text{ is a message}\]

Example 1

- \( \phi_1 = a, \text{enc}(a, b), b \)
- \( \phi_2 = a, \text{enc}(c, b), b \)
Static equivalence

Static equivalence: \( \phi \sim \phi' \)

Given two sequences of terms \( \phi, \phi' \), the intruder cannot distinguish them.

\[
\forall (\xi, \xi') \in \Pi^2, \xi \phi \downarrow = \xi' \phi \downarrow \iff \xi \phi' \downarrow = \xi' \phi' \downarrow \\
\forall \xi \in \Pi, \xi \phi \downarrow \text{ is a message} \iff \xi \phi' \downarrow \text{ is a message}
\]

Example 1

- \( \phi_1 = a, \text{enc}(a, b), b \)
- \( \phi_2 = a, \text{enc}(c, b), b \)

Example 2

- \( \phi_1 = a, \text{enc}(a, b) \)
- \( \phi_2 = a, \text{enc}(c, b) \)
Symbolic equivalence

\[ C \simeq_s C' \]

Given two constraint systems, any two associated traces are statically equivalent.

- for all \((\theta, \sigma) \in \text{Sol}(C)\), there exists \(\sigma'\) such that \((\theta, \sigma') \in \text{Sol}(C')\) and \(\phi\sigma \sim \phi'\sigma'\)
- for all \((\theta, \sigma') \in \text{Sol}(C')\), there exists \(\sigma\) such that \((\theta, \sigma) \in \text{Sol}(C)\), and \(\phi\sigma \sim \phi'\sigma'\)
Example 1

\[
\begin{align*}
A, B & \vdash x \\
A, B, \text{enc}(x, K) & \vdash \text{enc}(A, K)
\end{align*}
\]

\[
\begin{align*}
A, B & \vdash x \\
A, B, \text{enc}(A, K) & \vdash \text{enc}(A, K)
\end{align*}
\]
Example 1

\[ A, B \vdash x \]
\[ A, B, \text{enc}(x, K) \vdash \text{enc}(A, K) \]

\[ A, B \vdash x \]
\[ A, B, \text{enc}(A, K) \vdash \text{enc}(A, K) \]

Non-equivalent

The substitution of recipe \( \theta = \{ X_1 \mapsto ax_2, X_2 \mapsto ax_3 \} \) is only a solution for the first constraint system with \( \sigma = \{ x \mapsto B \} \), and
Example 2

\[ a, b, \text{enc}(n_a, k), \quad \vdash \text{enc}(x, k) \]
\[ a, b, \text{enc}(n_a, k), \text{enc}(\langle x, x \rangle, k), \quad k \quad \vdash \text{enc}(\langle n_a, n_a \rangle, k) \]

\[ a, b, \text{enc}(n_a, k), \quad \vdash \text{enc}(x, k) \]
\[ a, b, \text{enc}(n_a, k), \text{enc}(\langle x, x \rangle, k), \quad k' \quad \vdash \text{enc}(\langle n_a, n_a \rangle, k) \]
Example 2

\[ a, b, \text{enc}(n_a, k), \quad \vdash \text{enc}(x, k) \]
\[ a, b, \text{enc}(n_a, k), \text{enc}(\langle x, x \rangle, k), \quad k \quad \vdash \text{enc}(\langle n_a, n_a \rangle, k) \]

\[ a, b, \text{enc}(n_a, k), \quad \vdash \text{enc}(x, k) \]
\[ a, b, \text{enc}(n_a, k), \text{enc}(\langle x, \rangle, k), \quad k' \quad \vdash \text{enc}(\langle n_a, n_a \rangle, k) \]

Non-equivalent

- A solution: \( \sigma = \sigma' = \{x \mapsto n_a\} \), and
  \( \theta = \{X_1 \mapsto ax_3, X_2 \mapsto ax_4\} \)
- \( \phi\sigma \not\sim \phi'\sigma' : \xi = f(\text{dec}(ax_3, ax_5)), \xi' = \text{dec}(ax_4, ax_5) \)
1. Formalism
   - Constraint Systems
   - Equivalence

2. The rules
   - Definition and example
   - How to use them?

3. Termination, Completeness and Soundness
General idea

- Input: two constraint systems $C$ and $C'$
- Problem: is $C \approx_s C'$?
- Reduce the problem to a finite conjunction of constraint systems equivalence:
  \[ C_1 \approx_s C'_1 \land \ldots \land C_n \approx_s C'_n \]
- Decidability of each $C_i \approx_s C'_i$ has to be trivial
Constructor rule
Partition of the solution which ends or not by the application of a public constructor

\[ a, b \vdash \text{aenc}(x, p(a)) \]
Guessing from the top

**Constructor rule**

Partition of the solution which ends or not by the application of a public constructor

\[
\begin{align*}
& a, b \vdash \text{aenc}(x, p(a)) \\
\text{and} \\
& a, b \vdash x \\
& a, b \vdash p(a)
\end{align*}
\]
Constructor rule
Partition of the solution which ends or not by the application of a public constructor

\[
\begin{align*}
\text{a, b} & \vdash x \\
\text{a, b} & \vdash p(a) \\
\text{a, b} & \vdash \text{aenc}(x, p(a)) \\
\text{a, b} & \vdash \text{NoCons aenc}(x, p(a))
\end{align*}
\]
Guessing from the bottom

**Destructor rule**
Partition of the solution where a cypher can be decrypt or not.

\[ \text{aenc}(b, p(a)), \ a \vdash x \]
Guessing from the bottom

**Destructor rule**

Partition of the solution where a cypher can be decrypt or not.

\[
\text{IsDest} \
\text{aenc}(b, p(a)), a \vdash a \\
\text{IsDest} \
\text{aenc}(b, p(a)), a, b \vdash x
\]

\[\text{aenc}(b, p(a)), a \vdash x\]
Guessing from the bottom

Destructor rule

Partition of the solution where a cypher can be decrypt or not.

\[
\begin{align*}
\text{IsDest} & \quad \text{aenc}(b, p(a)), a \vdash a \\
\text{IsDest} & \quad \text{aenc}(b, p(a)), a, b \vdash x \\
\text{NoDest} & \quad \text{aenc}(b, p(a)), a \vdash x
\end{align*}
\]
Outline

1. Formalism
   - Constraint Systems
   - Equivalence

2. The rules
   - Definition and example
   - How to use them?

3. Termination, Completeness and Soundness
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B & \vdash \text{enc}(A, B) \\
C, D & \vdash \text{enc}(x, C)
\end{align*}
\]
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B &\vdash A \\
A, B &\vdash B \\
C, D &\vdash x \\
C, D &\vdash C
\end{align*}
\]

\[
\begin{align*}
A, B &\vdash \text{enc}(A, B) \\
C, D &\vdash \text{enc}(x, C)
\end{align*}
\]
Application of the rules on a constraint systems couple

\[\begin{align*}
A, B & \vdash A \\
A, B & \vdash B \\
C, D & \vdash x \\
C, D & \vdash C
\end{align*}\]

\[\begin{align*}
A, B & \vdash \text{enc}(A, B) \\
C, D & \vdash \text{enc}(x, C)
\end{align*}\]
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B &\vdash \text{enc}(A, B) \\
C, D &\vdash C
\end{align*}
\]
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B & \vdash A \\
A, B & \vdash B \\
\bot \\
A, B & \vdash \text{enc}(A, B) \\
C, D & \vdash C
\end{align*}
\]
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B & \vdash A \\
A, B & \vdash B \\
\bot
\end{align*}
\]

\[
\begin{align*}
A, B & \vdash \text{enc}(A, B) \\
C, D & \vdash C
\end{align*}
\]

\[
\begin{align*}
A, B & \vdash_{\text{NoCons}} \text{enc}(A, B) \\
C, D & \vdash_{\text{NoCons}} C
\end{align*}
\]
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B & \vdash \text{enc}(A, B) \\
C, D & \vdash x
\end{align*}
\]
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B & \vdash A \\
A, B & \vdash B \\
C, D & \vdash x_1 \\
C, D & \vdash x_2 \\
x & = \text{enc}(x_1, x_2)
\end{align*}
\]

\[
\begin{align*}
A, B & \vdash \text{enc}(A, B) \\
C, D & \vdash x
\end{align*}
\]
Application of the rules on a constraint systems couple

\[
\begin{align*}
A, B & \vdash A \\
A, B & \vdash B \\
C, D & \vdash x_1 \\
C, D & \vdash x_2 \\
x & = \text{enc}(x_1, x_2) \\
A, B & \vdash \text{NoCons} \ \text{enc}(A, B) \\
C, D & \vdash \text{NoCons} \ x
\end{align*}
\]
Theorem (Soundness)

If all leaves of a tree, whose root is labeled with \((C_0, C'_0)\) (a pair of initial constraints), are labeled either with \((\bot, \bot)\) or with some \((C, C')\) with \(C \neq \bot, C' \neq \bot\), then \(C_0 \approx_s C'_0\).
Soundness and completeness

**Theorem (Soundness)**

If all leaves of a tree, whose root is labeled with \((C_0, C'_0)\) (a pair of initial constraints), are labeled either with \((\perp, \perp)\) or with some \((C, C')\) with \(C \neq \perp\) and \(C' \neq \perp\), then \(C_0 \approx_s C'_0\).

**Theorem (Completeness)**

If \((C_0, C'_0)\) is a pair of initial constraints such that \(C_0 \approx_s C'_0\), then all leaves of a tree, whose root is labeled with \((C_0, C'_0)\), are labeled either with \((\perp, \perp)\) or with some \((C, C')\) with \(C \neq \perp\) and \(C' \neq \perp\).
Consider the initial pair of constraints \((C, C')\) given below:

\[
C = \begin{cases} 
  a \vdash \text{enc}(x_1, x_2) \\
  a, b \vdash x_1 
\end{cases} \\
C' = \begin{cases} 
  a \vdash y_1 \\
  a, b \vdash \text{enc}(y_1, y_2)
\end{cases}
\]
Termination problem

\[ C_1 = \begin{cases} 
  a \vdash x_1 \\
  a \vdash x_2 \\
  a, b \vdash x_1 
\end{cases} \]

\[ C'_1 = \begin{cases} 
  a \vdash z_1 \\
  a \vdash z_2 \\
  a, b \vdash \text{enc}(\text{enc}(z_1, z_2), y_2)
\end{cases} \]

with \( y_1 \overset{?}{=} \text{enc}(z_1, z_2) \)
Termination problem

\[
C_1 = \begin{cases} 
  a \vdash \text{enc}(t_1, t_2) \\
  a \vdash x_2 \\
  a, b \vdash t_1 \\
  a, b \vdash t_2 \\
\end{cases}
\]

with \( x_1 = \text{enc}(t_1, t_2) \)

\[
C_1' = \begin{cases} 
  a \vdash \text{z}_1 \\
  a \vdash \text{z}_2 \\
  a, b \vdash \text{enc}(\text{z}_1, \text{z}_2) \\
  a, b \vdash y_2 \\
\end{cases}
\]

with \( y_1 = \text{enc}(\text{z}_1, \text{z}_2) \)
Termination theorem

**Theorem**

There exists a strategy on the rules which terminates.
Demo
Future Works

Theory

1. Extension to non positive constraint systems (Ongoing work)
2. Extension to symbolic equivalence of constraint system set (Ongoing work)
3. Extension to trace equivalence of non deterministic protocol (Ongoing work)
4. Other cryptographic primitives

Implementation

1. Symbolic equivalence of positive constraint systems (Done)
2. Trace equivalence of positive protocol (Done but not efficient)