A decision procedure for proving observational equivalence

Vincent Cheval, Hubert Comon-Lundh, Stéphanie Delaune

LSV, Project SECSI

5 septembre 2009

Observational equivalence and security properties

Communicate on public channel with high security

Observational equivalence and security properties

Communicate on public channel with high security

Cryptography + Protocol (= Concurrent programs)

Communicate on public channel with high security

Cryptography + Protocol (= Concurrent programs)

How to be sure that there isn't any possible attack ?

Observational equivalence and security properties

Communicate on public channel with high security

Cryptography + Protocol (= Concurrent programs)

How to be sure that there isn't any possible attack ?

- Reliable cryptography
- Correct specification
- Implementation satisfying the specification

Observational equivalence and security properties

Communicate on public channel with high security

Cryptography + Protocol (= Concurrent programs)

How to be sure that there isn't any possible attack ?

- Reliable cryptography
- Correct specification
- Implementation satisfying the specification

Observational equivalence and security properties

Security properties

- Trace properties : simple secret, authentication,. . .
- Observational equivalence properties : strong secret, dictionnary attacks, anonymity. . .
- . . .

Observational equivalence and security properties

Security properties

- Trace properties : simple secret, authentication,. . .
- Observational equivalence properties : strong secret, dictionnary attacks, anonymity. . .
- . . .

Observational equivalence and security properties

Security properties

- Trace properties : simple secret, authentication,. . .
- Observational equivalence properties : strong secret, dictionnary attacks, anonymity. . .
- . . .

Definition (Observational equivalence)

$$P \approx Q \Leftrightarrow \forall R, P||R \text{ et } Q||R \text{ send the same signals}$$

Observational equivalence and security properties : Example

Handshake Protocol : Honest execution

$$\begin{array}{lll} 0. & A \longrightarrow B : & enc(M, k_{ab}) \\ 1. & B \longrightarrow A : & enc(f(M), k_{ab}) \end{array}$$

Observational equivalence and security properties : Example

Handshake Protocol : Honest execution

$$\begin{array}{lll} 0. & A \longrightarrow B : & enc(M, k_{ab}) \\ 1. & B \longrightarrow A : & enc(f(M), k_{ab}) \end{array}$$

Security properties : Off-line dictionnary attacks

After a finite number of sessions with the server, the intruder tries to guess the key by testing all the different possibilities.

Observational equivalence and security properties : Example

Handshake Protocol : Honest execution

$$\begin{array}{lll} 0. & A \longrightarrow B: & enc(M, k_{ab}) \\ 1. & B \longrightarrow A: & enc(f(M), k_{ab}) \end{array}$$

Security properties : Off-line dictionnary attacks

After a finite number of sessions with the server, the intruder tries to guess the key by testing all the different possibilities.

Formally

$$\nu k_{ab}.(P_A(k_{ab}) \| P_B(k_{ab}) \| P_A^2(k_{ab}) \| \ \ldots); c(k_{ab})$$
$$\approx$$
$$\nu k_{ab}.\nu k.(P_A(k_{ab}) \| P_B(k_{ab}) \| P_A^2(k_{ab}) \| \ \ldots); c(k)$$

Previous works

Huttel (2002)

- Only spi-calculus (fixed primitives)
- Untractable implementation (multi-exponential complexity)
- Doesn't handle trace properties.

Blanchet, Abadi, Fournet (2008)

- Infinite 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

Cortier, Delaune (2009) + Baudet (2005)

- Bounded number of sessions
- Observational equivalence between two deterministic positive processes
- Infinitely many traces are represented by constraint systems
- Observational equivalence of processes $\Leftrightarrow$ 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.

Our objectives

Objectives

- Find a new simpler algorithm for the decision of symbolic equivalence
  Reduction to the symbolic equivalence of the solved constraint systems of [CICZ 09]
- Algorithm for the decision of symbolic equivalence of solved constraint systems
- Implementation

Future works

- Extension to non-positive constraint systems.
- Extension to other equational theory and other crypographic primitives

Our objectives

Objectives

- Find a new simpler algorithm for the decision of symbolic equivalence
  Reduction to the symbolic equivalence of the solved constraint systems of [CICZ 09]
- Algorithm for the decision of symbolic equivalence of solved constraint systems
- Implementation

Future works

- Extension to non-positive constraint systems.
- Extension to other equational theory and other crypographic primitives

Our objectives

Objectives

- Find a new simpler algorithm for the decision of symbolic equivalence

  Reduction to the symbolic equivalence of the solved constraint systems of [CICZ 09]
- Algorithm for the decision of symbolic equivalence of solved constraint systems
- Implementation

Future works

- Extension to non-positive constraint systems.
- Extension to other equational theory and other crypographic primitives

Our objectives

Objectives

- Find a new simpler algorithm for the decision of symbolic equivalence
  Reduction to the symbolic equivalence of the solved constraint systems of [CICZ 09]
- Algorithm for the decision of symbolic equivalence of solved constraint systems
- Implementation

Future works

- Extension to non-positive constraint systems.
- Extension to other equational theory and other crypographic primitives

Constraint systems

Handshake Protocol : Honest execution

$$\begin{array}{lrl} 0. & A \longrightarrow B: & enc(M, k_{ab}) \\ 1. & B \longrightarrow A: & enc(f(M), k_{ab}) \end{array}$$

Constraint systems

Handshake Protocol : Honest execution

$$\begin{array}{lll} 0. & A \longrightarrow B: & enc(M, k_{ab}) \\ 1. & B \longrightarrow A: & enc(f(M), k_{ab}) \end{array}$$

Constraint system

$$enc(x, k_{ab})$$

Constraint systems

Handshake Protocol : Honest execution

$$\begin{array}{lll} 0. & A \longrightarrow B : & enc(M, k_{ab}) \\ 1. & B \longrightarrow A : & enc(f(M), k_{ab}) \end{array}$$

Constraint system

$$enc(M, k_{ab}) \quad \Vdash enc(x, k_{ab})$$

Constraint systems

Handshake Protocol : Honest execution

$$\begin{array}{lll} 0. & A \longrightarrow B : & enc(M, k_{ab}) \\ 1. & B \longrightarrow A : & enc(f(M), k_{ab}) \end{array}$$

Constraint system

$$\begin{array}{ll} enc(M, k_{ab}) & \Vdash enc(x, k_{ab}) \\ enc(M, k_{ab}), enc(f(x), k_{ab}) & \end{array}$$

Constraint systems

Handshake Protocol : Honest execution

$$\begin{array}{lll} 0. & A \longrightarrow B: & enc(M, k_{ab}) \\ 1. & B \longrightarrow A: & enc(f(M), k_{ab}) \end{array}$$

Constraint system

$$\begin{array}{ll} enc(M, k_{ab}) & \Vdash enc(x, k_{ab}) \\ enc(M, k_{ab}), enc(f(x), k_{ab}) & \Vdash enc(f(M), k_{ab}) \end{array}$$

Solution of a constraint system

Système de contrainte

$$
\begin{array}{ll}
enc(M, k_{ab}) & \Vdash enc(x, k_{ab}) \\
enc(M, k_{ab}), enc(f(x), k_{ab}) & \Vdash enc(f(M), k_{ab})
\end{array}
$$

Solution

- $\sigma = \{x \to M\}$
- $\xi_1 = ax_1$
- $\xi_2 = ax_2$

Symbolic equivalence

Static equivalence : $S \sim S'$

Given two term sequences S, S', the intruder cannot distinguish them.

$$\forall (\xi, \xi') \in \Pi^2, \xi[S]\downarrow = \xi'[S]\downarrow \Leftrightarrow \xi[S']\downarrow = \xi'[S']\downarrow$$

$(S, C) \approx_s (S', C')$

Given two constraint systems and two sequences, any two associated traces are staticly equivalent.

- $\forall (\sigma, \xi_1, \ldots, \xi_n) \in Sol(C), \exists \sigma' \text{ tq } (\sigma', \xi_1, \ldots, \xi_n) \in Sol(C') \wedge S\sigma \sim S'\sigma'$
- $\forall (\sigma', \xi_1, \ldots, \xi_n) \in Sol(C'), \exists \sigma \text{ tq } (\sigma, \xi_1, \ldots, \xi_n) \in Sol(C) \wedge S\sigma \sim S'\sigma'$

Example

Constraint system (Dictionnary attack)

$$
\begin{array}{ll}
enc(M, k_{ab}) & \Vdash enc(x, k_{ab}) \\
enc(M, k_{ab}), enc(f(x), k_{ab}) & \Vdash enc(f(M), k_{ab})
\end{array}
$$

$$
\begin{array}{l}
S = enc(M, k_{ab}), enc(f(x), k_{ab}), k_{ab} \\
S' = enc(M, k_{ab}), enc(f(x), k_{ab}), k
\end{array}
$$

Non-equivalent

- A solution : $\sigma = \{x \rightarrow M\}, \xi_1 = ax_1, \xi_2 = ax_2$

Example

Constraint system (Dictionnary attack)

$$\begin{array}{ll} enc(M, k_{ab}) & \Vdash enc(x, k_{ab}) \\ enc(M, k_{ab}), enc(f(x), k_{ab}) & \Vdash enc(f(M), k_{ab}) \end{array}$$

$$S = enc(M, k_{ab}), enc(f(x), k_{ab}), k_{ab}$$
$$S' = enc(M, k_{ab}), enc(f(x), k_{ab}), k$$

Non-equivalent

- A solution : $\sigma = \{x \to M\}, \xi_1 = ax_1, \xi_2 = ax_2$
- $S\sigma \not\sim S'\sigma$: $\xi = f(dec(ax_1, ax_3)), \xi' = dec(ax_2, ax_3)$

Reduction rules

Algorithm objectives

Decide the symbolic equivalence of constraint systems.

Reduction rules

Algorithm objectives

Decide the symbolic equivalence of constraint systems.

Contribution

Set of rules which :

- transforms the constraint systems into solved constraint systems

Reduction rules

Algorithm objectives

Decide the symbolic equivalence of constraint systems.

Contribution

Set of rules which :

- transforms the constraint systems into solved constraint systems
- preserves symbolic equivalence of constraint systems.

Reduction rules

Algorithm objectives

Decide the symbolic equivalence of constraint systems.

Contribution

Set of rules which :

- transforms the constraint systems into solved constraint systems
- preserves symbolic equivalence of constraint systems.
- terminates

Reduction rules

$$R_1 : \{\, T_i \Vdash f(t_1, t_2) \quad \nearrow \begin{cases} T_i \Vdash t_1 \\ T_i \Vdash t_2 \end{cases} \qquad \searrow \{\, T_i \nVdash f(t_1, t_2)$$

$$R_2 : \begin{cases} \mathcal{C}_0 \\ T_1, v, T_2 \Vdash u \\ \mathcal{C}_1 \end{cases} \quad \alpha = mgu(u, v) \quad \nearrow \begin{cases} \mathcal{C}_0\alpha \\ \mathcal{C}_1\alpha \end{cases} \qquad \searrow \begin{cases} \mathcal{C}_0 \\ T_1, v, T_2 \nVdash u \\ \mathcal{C}_1 \end{cases}$$

$$R_3 : \left\{ T_0, \{v_1\}_{v_2}, T_2 \Vdash u_1 \right. \quad \nearrow \left\{ \begin{array}{l} T_0, \{v_1\}_{v_2}, T_2 \quad \Vdash v_2 \\ T_0, \{v_1\}_{v_2}, T_2, v_1 \Vdash u_1 \end{array} \right.$$

$$\searrow \left\{ T_0, \{v_1\}_{v_2}, T_2 \Vdash u_1 \right.$$

$$R_4 : \left\{ T_1, \langle v_1, v_2 \rangle, T_2 \Vdash u_1 \right. \quad \longrightarrow \left\{ T_1, \langle v_1, v_2 \rangle, v_1, v_2, T_2 \Vdash u_1 \right.$$

Example

$$
\begin{array}{lcl}
d, e & \Vdash & <x, y> \\
d, e & \Vdash & z \\
d, e, z & \Vdash & w \\
d, e, z, \{c\}_{<d,e>} & \Vdash & c \\
d, e, z, \{c\}_{<d,e>}, \{z\}_h & \Vdash & \{\{d\}_e\}_h \\
d, e, z, \{c\}_{<d,e>}, \{z\}_h, \{e\}_h & \Vdash & \{w\}_h
\end{array}
$$

$$
\begin{array}{lcl}
a, b & \Vdash & <x, y> \\
a, b & \Vdash & z \\
a, b, \{a\}_b & \Vdash & w \\
a, b, \{a\}_b, \{c\}_{<a,b>} & \Vdash & c \\
a, b, \{a\}_b, \{c\}_{<a,b>}, \{z\}_f & \Vdash & \{\{a\}_b\}_f \\
a, b, \{a\}_b, \{c\}_{<a,b>}, \{z\}_f, \{b\}_f & \Vdash & \{w\}_f
\end{array}
$$

Example

$$
\begin{array}{lcl}
d, e & \Vdash & x \\
d, e & \Vdash & y \\
d, e & \Vdash & z \\
d, e, z & \Vdash & w \\
d, e, z, \{c\}_{<d,e>} & \Vdash & c \\
d, e, z, \{c\}_{<d,e>}, \{z\}_h & \Vdash & \{\{d\}_e\}\}_h \\
d, e, z, \{c\}_{<d,e>}, \{z\}_h, \{e\}_h & \Vdash & \{w\}_h
\end{array}
$$

$$
\begin{array}{lcl}
a, b & \Vdash & x \\
a, b & \Vdash & y \\
a, b & \Vdash & z \\
a, b, \{a\}_b & \Vdash & w \\
a, b, \{a\}_b, \{c\}_{<a,b>} & \Vdash & c \\
a, b, \{a\}_b, \{c\}_{<a,b>}, \{z\}_f & \Vdash & \{\{a\}_b\}_f \\
a, b, \{a\}_b, \{c\}_{<a,b>}, \{z\}_f, \{b\}_f & \Vdash & \{w\}_f
\end{array}
$$

Example

$$
\begin{array}{lll}
d, e & \Vdash & z \\
d, e, z & \Vdash & w \\
d, e, z, \{c\}_{<d,e>} & \Vdash & < d, e > \\
d, e, z, \{c\}_{<d,e>}, c & \Vdash & c \\
d, e, z, \{c\}_{<d,e>}, c, \{z\}_h & \Vdash & \{\{d\}_e\}_h \\
d, e, z, \{c\}_{<d,e>}, c, \{z\}_h, \{e\}_h & \Vdash & \{w\}_h
\end{array}
$$

$$
\begin{array}{lll}
a, b & \Vdash & z \\
a, b, \{a\}_b & \Vdash & w \\
a, b, \{a\}_b, \{c\}_{<a,b>} & \Vdash & < a, b > \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c & \Vdash & c \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{z\}_f & \Vdash & \{\{a\}_b\}_f \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{z\}_f, \{b\}_f & \Vdash & \{w\}_f
\end{array}
$$

Example

$$
\begin{array}{lcl}
d, e & \Vdash & z \\
d, e, z & \Vdash & w \\
d, e, z, \{c\}_{<d,e>} & \Vdash & < d, e > \\
d, e, z, \{c\}_{<d,e>}, c & \Vdash & c \\
d, e, z, \{c\}_{<d,e>}, c, \{z\}_h & \Vdash & \{\{d\}_e\}_h \\
d, e, z, \{c\}_{<d,e>}, c, \{z\}_h, \{e\}_h & \Vdash & \{w\}_h
\end{array}
$$

$$
\begin{array}{lcl}
a, b & \Vdash & \{a\}_b \\
a, b, \{a\}_b & \Vdash & w \\
a, b, \{a\}_b, \{c\}_{<a,b>} & \Vdash & < a, b > \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c & \Vdash & c \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f & \Vdash & \{\{a\}_b\}_f \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f, \{b\}_f & \Vdash & \{w\}_f
\end{array}
$$

Example

$$d, e \Vdash \{d\}_e$$
$$d, e, \{d\}_e \Vdash w$$
$$d, e, \{d\}_e, \{c\}_{<d,e>} \Vdash <d, e>$$
$$d, e, \{d\}_e, \{c\}_{<d,e>}, c \Vdash c$$
$$d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h \Vdash \{\{d\}_e\}_h$$
$$d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h, \{e\}_h \Vdash \{w\}_h$$

$$a, b \Vdash \{a\}_b$$
$$a, b, \{a\}_b \Vdash w$$
$$a, b, \{a\}_b, \{c\}_{<a,b>} \Vdash <a, b>$$
$$a, b, \{a\}_b, \{c\}_{<a,b>}, c \Vdash c$$
$$a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f \Vdash \{\{a\}_b\}_f$$
$$a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f, \{b\}_f \Vdash \{w\}_f$$

Example

$$
\begin{array}{lcl}
d, e & \Vdash & \{d\}_e \\
d, e, \{d\}_e & \Vdash & e \\
d, e, \{d\}_e, \{c\}_{<d,e>} & \Vdash & < d, e > \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c & \Vdash & c \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h & \Vdash & \{\{d\}_e\}_h \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h, \{e\}_h & \Vdash & \{e\}_h
\end{array}
$$

$$
\begin{array}{lcl}
a, b & \Vdash & \{a\}_b \\
a, b, \{a\}_b & \Vdash & b \\
a, b, \{a\}_b, \{c\}_{<a,b>} & \Vdash & < a, b > \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c & \Vdash & c \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f & \Vdash & \{\{a\}_b\}_f \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f, \{b\}_f & \Vdash & \{b\}_f
\end{array}
$$

Example

$$
\begin{array}{lcl}
d, e & \Vdash & \{d\}_e \\
d, e, \{d\}_e & \Vdash & w \\
d, e, \{d\}_e, \{c\}_{<d,e>} & \Vdash & < d, e > \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c & \Vdash & c \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h & \Vdash & \{\{d\}_e\}_h \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h, \{e\}_h & \Vdash & \{w\}_h
\end{array}
$$

$$
\begin{array}{lcl}
a, b & \Vdash & \{a\}_b \\
a, b, \{a\}_b & \Vdash & w \\
a, b, \{a\}_b, \{c\}_{<a,b>} & \Vdash & < a, b > \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c & \Vdash & c \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f & \Vdash & \{\{a\}_b\}_f \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f, \{b\}_f & \Vdash & \{w\}_f
\end{array}
$$

Example

$$
\begin{array}{lcl}
d, e & \Vdash & \{d\}_e \\
d, e, \{d\}_e & \Vdash & \{d\}_e \\
d, e, \{d\}_e, \{c\}_{<d,e>} & \Vdash & < d, e > \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c & \Vdash & c \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h & \Vdash & \{\{d\}_e\}_h \\
d, e, \{d\}_e, \{c\}_{<d,e>}, c, \{\{d\}_e\}_h, \{e\}_h & \Vdash & \{\{d\}_e\}_h
\end{array}
$$

$$
\begin{array}{lcl}
a, b & \Vdash & \{a\}_b \\
a, b, \{a\}_b & \Vdash & \{a\}_b \\
a, b, \{a\}_b, \{c\}_{<a,b>} & \Vdash & < a, b > \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c & \Vdash & c \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f & \Vdash & \{\{a\}_b\}_f \\
a, b, \{a\}_b, \{c\}_{<a,b>}, c, \{\{a\}_b\}_f, \{b\}_f & \Vdash & \{\{a\}_b\}_f
\end{array}
$$