Automated Verification of Selected Equivalences for Security Protocols

Bruno Blanchet
CNRS, École Normale Supérieure, Paris

Martín Abadi
University of California, Santa Cruz

Cédric Fournet
Microsoft Research, Cambridge

June 2005

Introduction

Analysis of cryptographic protocols:

- Powerful automatic tools for proving properties on behaviors (traces) of protocols (secrecy of keys, correspondences).

- Many important properties can be formalized as process equivalences, not as properties on behaviors:
  - secrecy of a boolean x in $P(x)$: $P(\text{true}) \approx P(\text{false})$
  - the process P implements an ideal specification Q: $P \approx Q$

  Equivalences are usually proved by difficult, long manual proofs.
  Already much research on this topic, using in particular sophisticated bisimulation techniques (e.g., Boreale et al).

Equivalences as properties of behaviors (1)

Goal: extend tools designed for proving properties of behaviors (here ProVerif) to the proof of process equivalences.

- We focus on equivalences between processes that differ only by the terms they contain, e.g., $P(\mathsf{true}) \approx P(\mathsf{false})$.

  Many interesting equivalences fall into this category.

- We introduce biprocesses to represent pairs of processes that differ only by the terms they contain.

  $P(\mathsf{true})$ and $P(\mathsf{false})$ are variants of a biprocess $P(\mathsf{diff}[\mathsf{true}, \mathsf{false}])$.

  The variants give a different interpretation to $\mathsf{diff}[\mathsf{true}, \mathsf{false}]$, true for the first variant, false for the second one.

Equivalences as properties of behaviors (2)

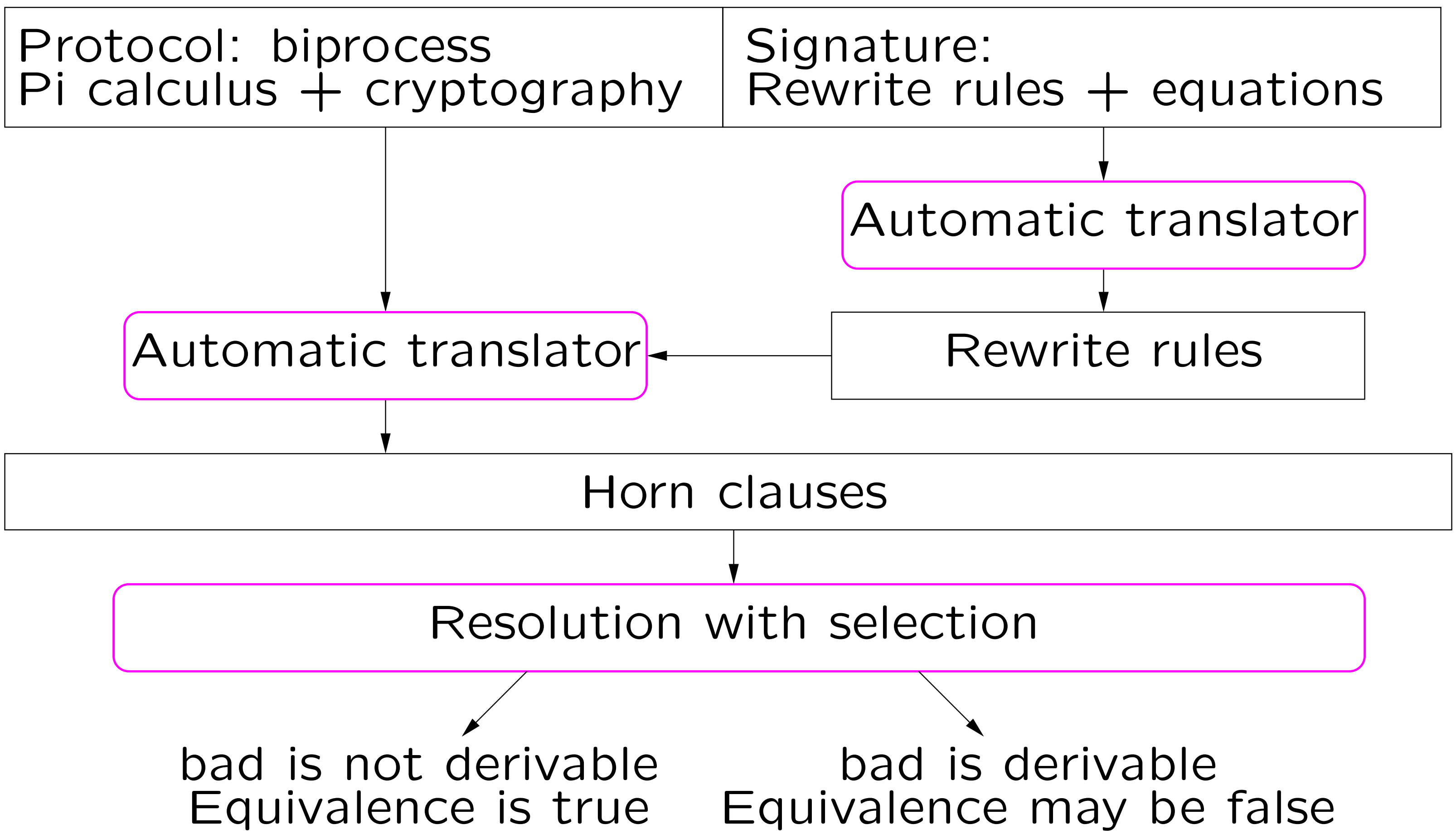
- We introduce a new operational semantics for biprocesses:

  A biprocess reduces when both variants reduce in the same way and after reduction, they still differ only by terms (so can be written using diff).

- We establish $P(\mathsf{true}) \approx P(\mathsf{false})$ by reasoning on behaviors of $P(\mathsf{diff}[\mathsf{true}, \mathsf{false}])$:

  If, for all reachable configurations, both variants reduce in the same way, then we have equivalence.

Overview of the verification method



The process calculus

Extension of the pi-calculus with function symbols for cryptographic primitives.

$M, N ::=$	terms
$\quad x, y, z$	variable
$\quad a, b, c, k, s$	name
$\quad f(M_1, \ldots, M_n)$	constructor application
$D ::=$	term evaluations
$\quad M$	term
$\quad$ eval $h(D_1, \ldots, D_n)$	function evaluation
$P, Q, R ::=$	processes
$\quad M(x).P$	input
$\quad \overline{M}\langle N \rangle.P$	output
$\quad \mathit{let}\ x = D\ \mathit{in}\ P\ \mathit{else}\ Q$	term evaluation
$\quad \mathbf{0} \quad P \mid Q \quad !P \quad (\nu a)P$	

Representation of cryptographic primitives

Two possible representations:

- When success/failure is visible: destructors with rewrite rules

  constructor $sencrypt$

  destructor $sdecrypt(sencrypt(x, y), y) \rightarrow x$

  The $else$ clause of the term evaluation is executed when no rewrite rule of some destructor applies.

- When success/failure is not visible: equations

  $sdecrypt(sencrypt(x, y), y) = x$

  $sencrypt(sdecrypt(x, y), y) = x$

The treatment of equations is one the main contributions of this work.

Semantics

$D \Downarrow M$ when the term evaluation D evaluates to M.
Uses rewrite rules of destructors and equations.

$\equiv$ transforms processes so that reduction rules can be applied.

Main reduction rules:

$\overline{N}\langle M \rangle.Q \mid N'(x).P \;\rightarrow\; Q \mid P\{M/x\}$ (Red I/O)
 if $\Sigma \vdash N = N'$

$let\ x = D\ in\ P\ else\ Q \rightarrow P\{M/x\}$ (Red Fun 1)
 if $D \Downarrow M$

$let\ x = D\ in\ P\ else\ Q \rightarrow Q$ (Red Fun 2)
 if there is no M such that $D \Downarrow M$

Observational equivalences and biprocesses

Two processes P and Q are observationally equivalent ($P \approx Q$) when the adversary cannot distinguish them.

A biprocess P is a process with diff.

$\mathsf{fst}(P)$ = the process obtained by replacing $\mathsf{diff}[M, M']$ with M.

$\mathsf{snd}(P)$ = the process obtained by replacing $\mathsf{diff}[M, M']$ with M'.

P satisfies observational equivalence when $\mathsf{fst}(P) \approx \mathsf{snd}(P)$.

Semantics of biprocesses

A biprocess reduces when both variants of the process reduce in the same way.

$$\overline{N}\langle M\rangle.Q \mid N'(x).P \;\rightarrow\; Q \mid P\{M/x\} \qquad \text{(Red I/O)}$$

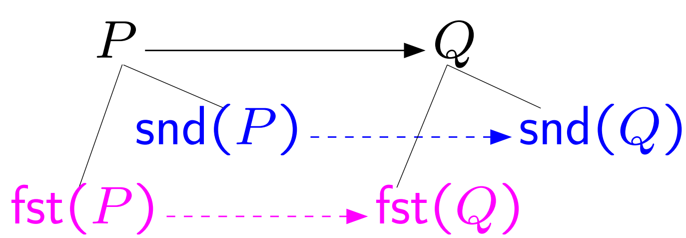
if $\Sigma \vdash \mathsf{fst}(N) = \mathsf{fst}(N')$ and $\Sigma \vdash \mathsf{snd}(N) = \mathsf{snd}(N')$

$$let\ x = D\ in\ P\ else\ Q \rightarrow P\{\mathsf{diff}[M_1, M_2]/x\} \qquad \text{(Red Fun 1)}$$

if $\mathsf{fst}(D) \Downarrow M_1$ and $\mathsf{snd}(D) \Downarrow M_2$

$$let\ x = D\ in\ P\ else\ Q \rightarrow Q \qquad \text{(Red Fun 2)}$$

if there is no M_1 such that $\mathsf{fst}(D) \Downarrow M_1$ and there is no M_2 such that $\mathsf{snd}(D) \Downarrow M_2$



Proof of observational equivalence using biprocesses

Let P_0 be a closed biprocess.

If for all configurations P reachable from P_0 (in the presence of an adversary), both variants of P reduce in the same way, then P_0 satisfies observational equivalence.

Formalizing the adversary

Let P_0 be a closed biprocess.

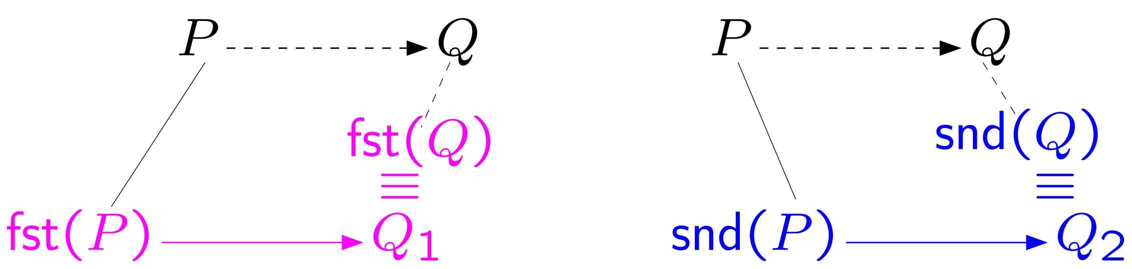
> If for all configurations P reachable from P_0 (in the presence of an adversary), both variants of P reduce in the same way, then P_0 satisfies observational equivalence.

An adversary is represented by a plain evaluation context (evaluation context without diff), so:

> If, for all plain evaluation contexts C and reductions $C[P_0] \rightarrow^* P$, both variants of P reduce in the same way, then P_0 satisfies observational equivalence.

Formalizing "reduce in the same way"

The biprocess P is uniform when
$\mathsf{fst}(P) \rightarrow Q_1$ implies $P \rightarrow Q$ for some biprocess Q with $\mathsf{fst}(Q) \equiv Q_1$,
and symmetrically for $\mathsf{snd}(P) \rightarrow Q_2$.



If, for all plain evaluation contexts C and reductions $C[P_0] \rightarrow^* P$,
the biprocess P is uniform,
then P_0 satisfies observational equivalence.

Result

Let P_0 be a closed biprocess.

Suppose that, for all plain evaluation contexts C, all evaluation contexts C', and all reductions $C[P_0] \rightarrow^* P$,

1. the (Red I/O) rules apply in the same way on both variants.

   if $P \equiv C'[\overline{N}\langle M\rangle.Q \mid N'(x).R]$, then $\Sigma \vdash \mathsf{fst}(N) = \mathsf{fst}(N')$ if and only if $\Sigma \vdash \mathsf{snd}(N) = \mathsf{snd}(N')$,

2. the (Red Fun) rules apply in the same way on both variants.

   if $P \equiv C'[let\ x = D\ in\ Q\ else\ R]$, then there exists M_1 such that $\mathsf{fst}(D) \Downarrow M_1$ if and only if there exists M_2 such that $\mathsf{snd}(D) \Downarrow M_2$.

Then P_0 satisfies observational equivalence.

Example: Non-deterministic encryption

Non-deterministic public-key encryption is modeled by an equation:

$$dec(enc(x, pk(s), a), s) = x$$

Without knowledge of the decryption key, ciphertexts appear to be unrelated to the plaintexts.

Ciphertexts are indistinguishable from fresh names:

$$(\nu s)(\overline{c}\langle pk(s)\rangle \mid !c'(x).(\nu a)\overline{c}\langle \mathsf{diff}[enc(x, pk(s), a), a]\rangle)$$

satisfies equivalence.

This equivalence can be proved using the previous result, and verified automatically by ProVerif.

Treatment of equations

We automatically transform equations into rewrite rules, much easier to handle (and already handled in ProVerif), e.g., transform $\mathrm{g}\hat{\ }x\hat{\ }y = \mathrm{g}\hat{\ }y\hat{\ }x$ to $\mathrm{g}\hat{\ }x\hat{\ }y \rightarrow \mathrm{g}\hat{\ }y\hat{\ }x$.

We have shown that, for each trace with equations, there is a corresponding trace with rewrite rules, and conversely.

Then we obtain a result for proving equivalences using rewrite rules instead of equations.

(See formal details in the paper.)

Translation into clauses

As in our previous work, we translate the protocol and the adversary into a set of Horn clauses.

The predicates differ in order to translate behaviors of biprocesses instead of processes:

$F ::=$	facts
$\text{att}'(p, p')$	the attacker has p (resp. p')
$\text{msg}'(p_1, p_2, p'_1, p'_2)$	message p_2 is sent on channel p_1 (resp. p'_2 on p'_1)
$\text{input}'(p, p')$	input on p (resp. p')
$\text{nounif}(p, p')$	p and p' do not unify modulo Σ
bad	the property may be false

Magenta arguments for the first version of the biprocess, blue ones for the second version.

Example: some generated clauses

The biprocess of the non-deterministic encryption example:

$$(\nu s)(\overline{c}\langle pk(s)\rangle \mid !c'(x).(\nu a)\overline{c}\langle \mathsf{diff}[enc(x, pk(s), a), a]\rangle)$$

yields the clauses:

$$\mathsf{msg}'(c, pk(s), c, pk(s))$$
$$\mathsf{msg}'(c', x, c', x') \rightarrow \mathsf{msg}'(c, enc(x, pk(s), a[i, x]), c, a[i, x'])$$

The first clause corresponds to the output of the public key $pk(s)$.

The second clause corresponds to the other output.

Resolution algorithm

Theorem 1 *If* bad *is not a logical consequence of the clauses, then* P_0 *satisfies observational equivalence.*

We determine whether bad is a logical consequence of the clauses using a resolution-based algorithm.

This algorithm uses domain-specific simplification steps (for predicate nounif in particular, using unification modulo the equational theory of Σ).

Applications

- Weak secrets: We can express that a password is protected against off-line guessing attacks by an equivalence, and prove it using our technique (done for 4 versions of EKE).

- Authenticity: We can formalize authenticity as an equivalence and prove it (for the Wide-Mouth Frog protocol).

- JFK: We can show that the encrypted messages of JFK are equivalent to fresh names, with our technique plus the property that observational equivalence is contextual.

Total runtime: 45 s on a Pentium M 1.8 GHz.

Conclusion

Contributions:

- Fully automatic proof of some process equivalences.
- Treatment of cryptographic primitives represented by equations.

Implementation and more information at

`http://www.di.ens.fr/~blanchet/obsequi/`