Communication security: Formal models and proofs

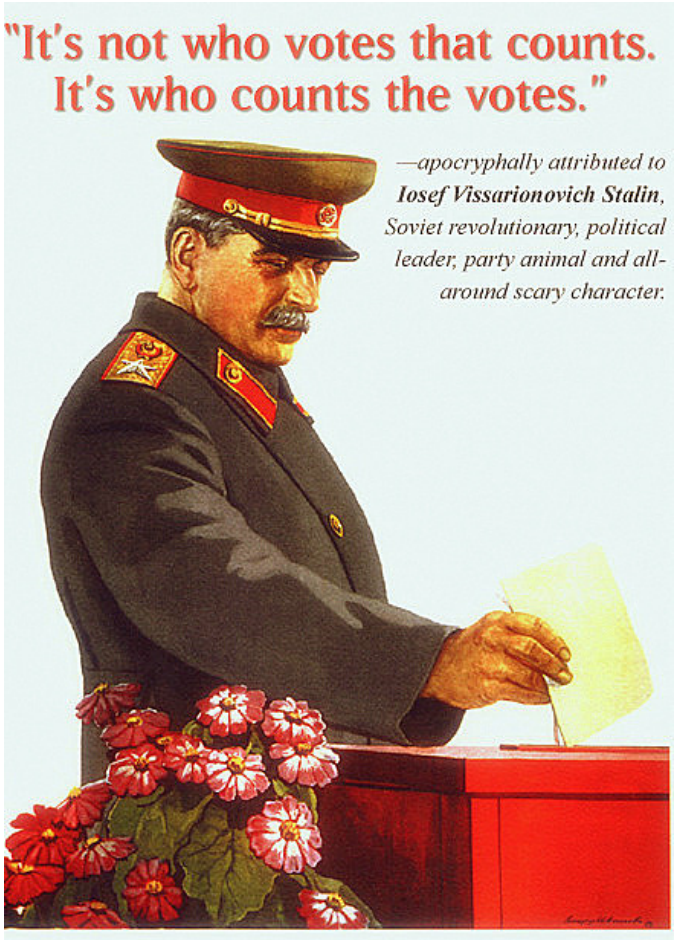
Hubert Comon

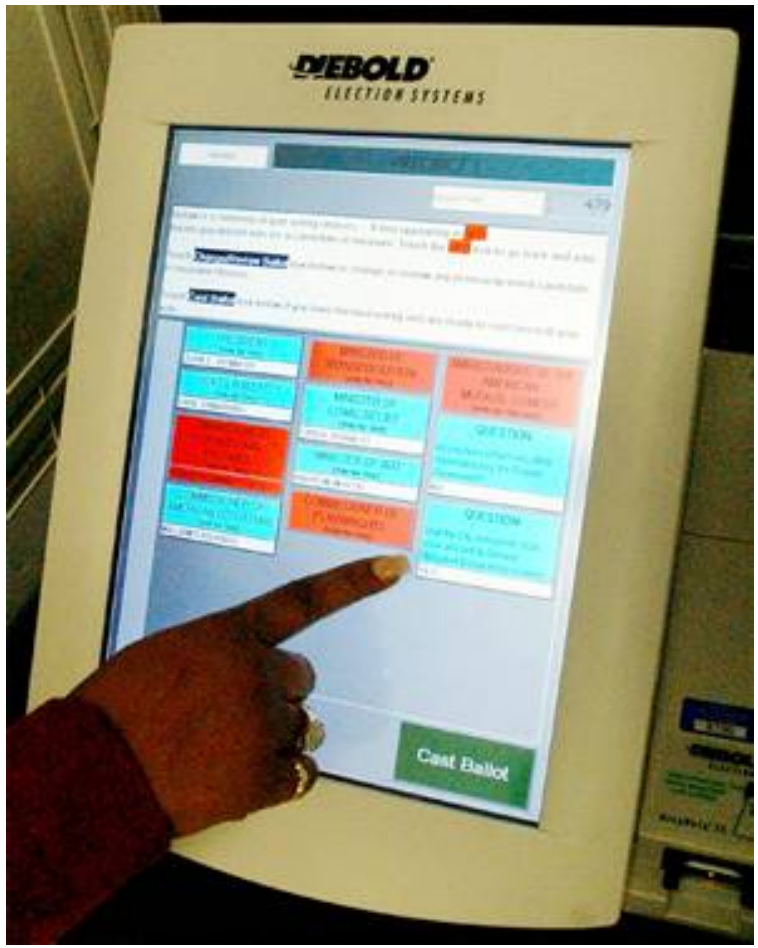
September 14, 2016

The context (I)

- credit cards
- contactless cards
- telephones
- online transactions
- cars, fridges,... Internet of Things
- Big Brother: NSA
- Biomedical applications
- ...

The context (II)





The context (III)

- Security protocols
- Testing is not very useful
- Hiding the code is not a good idea
- The scope of formal methods

A simple handshake protocol

$$
\begin{aligned}
A \to B : &\quad \nu n, r.\ \mathsf{aenc}(\langle A, n \rangle, \mathsf{pk}(\mathsf{sk}_B), r) \\
B \to A : &\quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')
\end{aligned}
$$

The formal verification problem

$$\forall \mathcal{A}. \quad \mathcal{A} \parallel P \quad \models \quad \phi$$

The formal verification problem

$$\forall \mathcal{A}. \quad \mathcal{A} \parallel P \models \phi$$

$$\forall \mathcal{A}. \quad \mathcal{A} \parallel P_1 \sim \mathcal{A} \parallel P_2$$

The formal verification problem

$$\forall \mathcal{A}. \quad \mathcal{A} \parallel P \quad \models \quad \phi$$

$$\forall \mathcal{A}. \quad \mathcal{A} \parallel P_1 \quad \sim \quad \mathcal{A} \parallel P_2$$

Universal quantification on $\mathcal{A}$: we cannot apply directly model-checking techniques.

The formal verification problem

$$\forall \mathcal{A}. \quad \mathcal{A} \parallel P \quad \models \quad \phi$$

$$\forall \mathcal{A}. \quad \mathcal{A} \parallel P_1 \quad \sim \quad \mathcal{A} \parallel P_2$$

Universal quantification on $\mathcal{A}$: we cannot apply directly model-checking techniques.

One important issue: range of $\mathcal{A}$?

Attacker models

Attacker models

The DY-attacker

Messages are terms, the attacker is defined through an equation theory or an inference system

Attacker models

The DY-attacker

Messages are terms, the attacker is defined through an equation theory or an inference system

The computational attacker

Messages are bitstrings, the attacker is a probabilistic polynomial time Turing machine

Attacker models

The DY-attacker

Messages are terms, the attacker is defined through an equation theory or an inference system

The computational attacker

Messages are bitstrings, the attacker is a probabilistic polynomial time Turing machine

Other attackers

Goals of the lecture

Verification inputs

- Cryptographic libraries
- Protocol programs
- Attacker model
- Security property

Goals of the lecture

Verification inputs

- Cryptographic libraries
- Protocol programs
- Attacker model
- Security property

Goals of the lecture

Show how to derive the proof obligations in a parametric way, abstracting from crypto libraries, attacker models.
Focus on the semantics of protocols, for arbitrary libraries and attacker models.

Roadmap

4 successive versions of the calculus, by increasing expressiveness (we could have considered the last case only...)

1. Simple case
2. Adding events: required for agreement properties
3. Adding replication
4. Adding channel generation: required for computational semantics

Then indistinguishability properties (privacy).

Cryptographic libraries

Syntax

- An arbitrary set of cryptographic primitives $\mathcal{F}$: hash, public-key encryption(s), symmetric encryption(s), zkp,... represented by (typed) function symbols
- At least one random generation algorithm. Random numbers are represented by *names* $n, n_1, r, \ldots$ out of a set $\mathcal{N}$

Terms are built over variables, function symbols and names.

Cryptographic libraries

Semantics

$\mathcal{M}$ is an interpretation domain. Typically ground or constructor terms (the DY semantics) or bitstrings (the computational semantics).
$\mathcal{M}$ includes error messages (exceptions) $\texttt{Err}$.
If σ is an environment (mapping from variables to $\mathcal{M}$), u is a term,

$$[\![u]\!]_\sigma^{\mathcal{M}}$$

is the interpretation of u in $\mathcal{M}$ w.r.t. σ: $\mathcal{M}$ is a (partial) $\mathcal{F}$-algebra.
The interpretation is strict:

$$u_i \in \texttt{Err} \quad \Rightarrow \quad [\![f(u_1, \ldots, u_n)]\!]_\sigma^{\mathcal{M}} \in \texttt{Err}$$

Cryptographic libraries

A possible set of function symbols

- $\mathrm{aenc}(u, pk, r)$ is (supposed to be) the asymmetric encryption of u with the public key pk and random input r.
- $\mathrm{dec}(u, \mathsf{sk})$ is (supposed to be) the decryption of u with the secret key sk
- $\mathrm{pk}(\mathsf{sk})$ is (supposed to be) the public key associated with the secret key sk
- $\langle u, v \rangle$
- $\pi_1(u), \pi_2(u)$

Cryptographic libraries

A DY model

$\mathcal{M}_{DY}$ (messages) is the least set of ground terms such that:

- $\mathcal{N} \subseteq \mathcal{M}_{DY}$
- if $u, v \in \mathcal{M}_{DY}$ then $\langle u, v \rangle \in \mathcal{M}_{DY}$
- if $k \in \mathcal{N}$ then $\mathsf{pk}(k) \in \mathcal{M}_{DY}$
- if $u \in \mathcal{M}_{DY}, k, r \in \mathcal{N}$, then $\mathsf{aenc}(u, \mathsf{pk}(k), r) \in \mathcal{M}_{DY}$.

$\mathcal{M}_{DY}$ also includes special error terms `Err` (not messages).

$$\begin{array}{rcll} \mathsf{dec}(\mathsf{aenc}(u, \mathsf{pk}(k), r), k) & \to & u & \text{For } k, r \in \mathcal{N},\ u \text{ a message} \\ \pi_1(\langle u, v \rangle) & \to & u & u, v \text{ are messages} \\ \pi_2(\langle u, v \rangle) & \to & v & u, v \text{ are messages} \end{array}$$

$$[\![u]\!]_\sigma^{\mathcal{M}_{DY}} = u\sigma\downarrow$$

Any irreducible ground term, which is not a message, is an error.

Cryptographic libraries

Computational models

- $\eta \in \mathbb{N}$ is a security parameter
- τ maps $\mathcal{N}$ to $\{0,1\}^\eta$
- $\mathcal{M}_c(\tau, \eta) \subseteq \{0,1\}^*$
- $[\![n]\!]^{\mathcal{M}_c(\tau,\eta)} = \tau(n)$
- $\mathsf{aenc}(\_,\_,\_), \mathsf{dec}(\_,\_), \mathsf{pk}(\_)$ are interpreted as a public-key encryption scheme.
- with an interpretation of pairing/projections, $\mathcal{M}_c(\tau, \eta)$ is an $\mathcal{F}$-algebra

A simple process calculus

Syntax

$P ::=$		0	null process (stalled)
	$\mid$	$\mathsf{in}(x).P$	input of x (binds x)
	$\mid$	$\mathsf{out}(t).P$	output of t
	$\mid$	if $\mathtt{EQ}(u, v)$ then P else P	conditional branching
	$\mid$	let $y = u$ in P	evaluation (binds y)
	$\mid$	$\nu \overline{n}.P$	random generation
	$\mid$	$P \Vert P$	parallel composition

All variable occurrences are bound.

Example

The simple handshake protocol

$$A \to B : \quad \nu n, r.\ \mathsf{aenc}(\langle A, n \rangle, \mathsf{pk}(\mathsf{sk}_B), r)$$
$$B \to A : \quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')$$

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) =$$

$$B(\mathsf{sk}_b) =$$

Example

The simple handshake protocol

$$A \to B : \quad \nu n, r.\ \mathsf{aenc}(\langle A, n\rangle, \mathsf{pk}(\mathsf{sk}_B), r)$$
$$B \to A : \quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')$$

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) = \nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r)).$$

$$B(\mathsf{sk}_b) =$$

Example

The simple handshake protocol

$$A \to B : \quad \nu n, r.\ \mathsf{aenc}(\langle A, n\rangle, \mathsf{pk}(\mathsf{sk}_B), r)$$
$$B \to A : \quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')$$

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) = \nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r)).$$

$$B(\mathsf{sk}_b) = \nu r'.\ \mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}$$

Example

The simple handshake protocol

$$A \to B : \quad \nu n, r.\ \mathsf{aenc}(\langle A, n\rangle, \mathsf{pk}(\mathsf{sk}_B), r)$$
$$B \to A : \quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')$$

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) = \nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r)).$$

$$B(\mathsf{sk}_b) = \nu r'.\ \mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r')).\ \mathbf{0}.$$

Example

The simple handshake protocol

$$A \to B : \quad \nu n, r.\ \mathsf{aenc}(\langle A, n\rangle, \mathsf{pk}(\mathsf{sk}_B), r)$$
$$B \to A : \quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')$$

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) = \nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r)).\ \mathsf{in}(z).\ \mathsf{let}\ z_1 = \mathsf{dec}(z, \mathsf{sk}_a)\ \mathsf{in}$$

$$B(\mathsf{sk}_b) = \nu r'.\ \mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r')).\ \mathbf{0}.$$

Example

The simple handshake protocol

$$A \to B : \quad \nu n, r.\ \mathsf{aenc}(\langle A, n\rangle, \mathsf{pk}(\mathsf{sk}_B), r)$$
$$B \to A : \quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')$$

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) = \nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r)).\ \mathsf{in}(z).\ \mathsf{let}\ z_1 = \mathsf{dec}(z, \mathsf{sk}_a)\ \mathsf{in}\ \mathsf{if}\ \mathtt{EQ}(z_1, n)\ \mathsf{then}\ \mathbf{0}(\mathit{Success})\ \mathsf{else}\ \mathbf{0}(\mathit{Fail})$$

$$B(\mathsf{sk}_b) = \nu r'.\ \mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r')).\ \mathbf{0}.$$

Example

The simple handshake protocol

$$A \to B : \quad \nu n, r.\ \mathsf{aenc}(\langle A, n\rangle, \mathsf{pk}(\mathsf{sk}_B), r)$$
$$B \to A : \quad \nu r'.\ \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r')$$

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) = \nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r)).\ \mathsf{in}(z).\ \mathsf{let}\ z_1 = \mathsf{dec}(z, \mathsf{sk}_a)\ \mathsf{in}\ \mathsf{if}\ \mathtt{EQ}(z_1, n)\ \mathsf{then}\ \mathbf{0}(\mathit{Success})\ \mathsf{else}\ \mathbf{0}(\mathit{Fail})$$

$$B(\mathsf{sk}_b) = \nu r'.\ \mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r')).\ \mathbf{0}.$$

$$\nu \mathsf{sk}_a, \mathsf{sk}_b.\quad \mathsf{out}(\langle \mathsf{pk}(\mathsf{sk}_a), \mathsf{pk}(\mathsf{sk}_b)\rangle).\quad (A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) \parallel B(\mathsf{sk}_b))$$

Structural equivalence

$$
\begin{aligned}
0 \parallel P &\equiv P \\
P \parallel Q &\equiv Q \parallel P \\
P \parallel (Q \parallel R) &\equiv (P \parallel Q) \parallel R \\
\nu n.P &\equiv \nu n'.P\{n \mapsto n'\} \\
\mathsf{in}(x).P &\equiv \mathsf{in}(x').P\{x \mapsto x'\} \\
\mathsf{let}\ x = u\ \mathsf{in}\ P &\equiv \mathsf{let}\ x' = u\ \mathsf{in}\ P\{x \mapsto x'\} \\
(\nu n.P) \parallel Q &\equiv \nu n'.(P \parallel Q) \qquad \text{if } n \notin \mathsf{freenames}(Q)
\end{aligned}
$$

Operational semantics

States of the network are tuples (ϕ, σ, P), where

- ϕ is a *frame* of the form $\nu\overline{n}.m_1, \ldots, m_k$, where $\overline{n}$ is a set of names (used so far) and $m_1, \ldots, m_k$ is a sequence of values in $\mathcal{M}$ (that have been sent out so far)
- σ is an environment: an assignment of the free variables to values in $\mathcal{M}$
- P is a process

The semantics is a labeled transition system, whose labels are the inputs provided by the attacker (sometimes, an empty input)

Operational semantics

The transition system (I)

$$\frac{}{(\phi, \sigma, \mathsf{in}(x).P) \xrightarrow{u} (\phi, \sigma \uplus \{x \mapsto u\}, P)}$$

Operational semantics

The transition system (I)

$$\frac{}{(\phi, \sigma, \mathsf{in}(x).P) \xrightarrow{u} (\phi, \sigma \uplus \{x \mapsto u\}, P)}$$

$$\frac{(\phi, \sigma, P) \xrightarrow{u} (\phi', \sigma', P')}{(\phi, \sigma, \mathsf{if}\ \mathtt{EQ}(s, t)\ \mathsf{then}\ P\ \mathsf{else}\ Q) \xrightarrow{u} (\phi', \sigma', P')}$$

$\mathsf{if} [\![s]\!]^{\mathcal{M}}_{\sigma} = [\![t]\!]^{\mathcal{M}}_{\sigma} \notin \mathtt{Err}$

Operational semantics

The transition system (I)

$$\frac{}{(\phi, \sigma, \mathsf{in}(x).P) \xrightarrow{u} (\phi, \sigma \uplus \{x \mapsto u\}, P)}$$

$$\frac{(\phi, \sigma, P) \xrightarrow{u} (\phi', \sigma', P')}{(\phi, \sigma, \mathsf{if}\ \mathtt{EQ}(s,t)\ \mathsf{then}\ P\ \mathsf{else}\ Q) \xrightarrow{u} (\phi', \sigma', P')}$$

$\mathsf{if} [\![s]\!]^{\mathcal{M}}_{\sigma} = [\![t]\!]^{\mathcal{M}}_{\sigma} \notin \mathtt{Err}$

$$\frac{(\phi, \sigma, Q) \xrightarrow{u} (\phi', \sigma', P')}{(\phi, \sigma, \mathsf{if}\ \mathtt{EQ}(s,t)\ \mathsf{then}\ P\ \mathsf{else}\ Q) \xrightarrow{u} (\phi', \sigma', P')}$$

$\mathsf{if} [\![s]\!]^{\mathcal{M}}_{\sigma} \neq [\![t]\!]^{\mathcal{M}}_{\sigma}$ or $[\![s]\!]^{\mathcal{M}}_{\sigma} \in \mathtt{Err}$ or $[\![t]\!]^{\mathcal{M}}_{\sigma} \in \mathtt{Err}$

Operational semantics

The transition system (II)

$$\frac{}{(\phi, \sigma, \text{let } x = u \text{ in } P) \;\rightarrow\; (\phi, \sigma \uplus \{x \mapsto w\}, P)} \;\texttt{if}\; [\![u]\!]_\sigma^{\mathcal{M}} = w \notin \texttt{Err}$$

Operational semantics

The transition system (II)

$$\frac{}{(\phi, \sigma, \mathsf{let}\ x = u\ \mathsf{in}\ P) \ \rightarrow\ (\phi, \sigma \uplus \{x \mapsto w\}, P)}\ \texttt{if}\ [\![u]\!]^{\mathcal{M}}_{\sigma} = w \notin \texttt{Err}$$

$$\frac{}{(\nu \overline{n}.\theta, \sigma, \mathsf{out}(s).P) \ \rightarrow\ (\nu \overline{n}.\theta \cdot [\![s]\!]^{\mathcal{M}}_{\sigma}, \sigma, P)}$$

Operational semantics

The transition system (II)

$$\frac{}{(\phi, \sigma, \mathsf{let}\ x = u\ \mathsf{in}\ P) \ \rightarrow\ (\phi, \sigma \uplus \{x \mapsto w\}, P)}\ \mathtt{if}\ [\![u]\!]^{\mathcal{M}}_{\sigma} = w \notin \mathtt{Err}$$

$$\frac{}{(\nu \overline{n}.\theta, \sigma, \mathsf{out}(s).P) \ \rightarrow\ (\nu \overline{n}.\theta \cdot [\![s]\!]^{\mathcal{M}}_{\sigma}, \sigma, P)}$$

$$\frac{(\phi, \sigma, P) \ \xrightarrow{u}\ (\phi', \sigma', P')}{(\phi, \sigma, P \| Q) \ \xrightarrow{u}\ (\phi', \sigma', P' \| Q)}$$

Operational semantics

The transition system (II)

$$\frac{}{(\phi, \sigma, \mathsf{let}\ x = u\ \mathsf{in}\ P) \ \rightarrow\ (\phi, \sigma \uplus \{x \mapsto w\}, P)}\ \mathtt{if}\ [\![u]\!]^{\mathcal{M}}_{\sigma} = w \notin \mathtt{Err}$$

$$\frac{}{(\nu\overline{n}.\theta, \sigma, \mathsf{out}(s).P) \ \rightarrow\ (\nu\overline{n}.\theta \cdot [\![s]\!]^{\mathcal{M}}_{\sigma}, \sigma, P)}$$

$$\frac{(\phi, \sigma, P) \ \xrightarrow{u}\ (\phi', \sigma', P')}{(\phi, \sigma, P \| Q) \ \xrightarrow{u}\ (\phi', \sigma', P' \| Q)}$$

$$\frac{}{(\nu\overline{n}.\theta, \sigma, \nu n.P) \ \rightarrow\ \nu\overline{n} \uplus n.\theta, \sigma, P)}\ \mathsf{if}\ n \notin \overline{n} \cup \mathsf{freename}(\theta)$$

Example

On the black board.

Restricting the feasible transitions

$$(\phi_1, \sigma_1, P_1) \xrightarrow{u_1} \cdots \xrightarrow{u_{k-1}} (\phi_k, \sigma_k, P_k)$$

is possible w.r.t. model $\mathcal{M}$ and an attacker $\mathcal{A}$ if , for every i,

$$\mathcal{A}([\![\phi_i]\!]^{\mathcal{M}}_{\sigma_i}, P_i) = [\![u_i]\!]^{\mathcal{M}}_{\sigma_i}$$

Note: could include a state in $\mathcal{A}$.

Example DY

There is a DY attacker $\mathcal{A}$ such that $\mathcal{A}(\phi) = [\![u]\!]_\sigma^{\mathcal{M}_{DY}}$ iff

$$\phi \vdash_I u\sigma \downarrow$$

where I is defined by:

$$\frac{\phi \vdash u_1 \cdots \phi \vdash u_n}{\phi \vdash f(u_1, \ldots, u_n) \downarrow}$$

For every $f \in \mathcal{F}$

$$\frac{}{\nu \overline{n}.u_1, \ldots, u_n \vdash u_i}$$

$$\frac{}{\nu \overline{n}.\theta \vdash n'}$$

if $n' \in \mathcal{N} \setminus \overline{n}$.

Exercise

In the simple handshake example, describe all feasible transition sequences in the DY model (assume the name extrusion, let, conditionals and outputs are always performed before inputs).
Is the nonce n secret ?

Example computational

$\mathcal{A}$ is a Probabilistic Polynomial Time Turing machine (PPT).
Some inputs that were not possible in the DY model might now be possible.

A typical example

$\mathcal{A}$ might be able to compute (with a significant probability) $[\![\mathsf{aenc}(u, \mathsf{pk}(k_1), r_1)]\!]^{\mathcal{M}_c(\tau,\eta)}$ from $[\![\mathsf{aenc}(v, \mathsf{pk}(k_1), r_1)]\!]^{\mathcal{M}_c(\tau,\eta)}$

Example computational

$\mathcal{A}$ is a Probabilistic Polynomial Time Turing machine (PPT).
Some inputs that were not possible in the DY model might now be possible.

A typical example

$\mathcal{A}$ might be able to compute (with a significant probability) $[\![\mathsf{aenc}(u, \mathsf{pk}(k_1), r_1)]\!]^{\mathcal{M}_c(\tau,\eta)}$ from $[\![\mathsf{aenc}(v, \mathsf{pk}(k_1), r_1)]\!]^{\mathcal{M}_c(\tau,\eta)}$

$$\exists \mathcal{A}, \quad \mathbf{Prob}\{\tau, \rho : \mathcal{A}([\![\mathsf{aenc}(v, \mathsf{pk}(k_1), r_1)]\!]^{\mathcal{M}_c(\tau,\eta)}) = [\![\mathsf{aenc}(u, \mathsf{pk}(k_1), r_1)]\!]^{\mathcal{M}_c(\tau,\eta)}\} > \epsilon(\eta)$$

ϵ is *non-negligible*: there is a polynomial Pol such that

$$\liminf_{\eta \to +\infty} \epsilon(\eta) \times \mathrm{Pol}(\eta) > 1$$

Confidentiality

In the DY case

Is there a DY attacker $\mathcal{A}$ and a feasible transition sequence

$$(\emptyset, \emptyset, P) \xrightarrow{*} (\phi, \sigma, Q)$$

such that $\mathcal{A}(\phi, Q) = s$?

Confidentiality

In the DY case

Is there a DY attacker $\mathcal{A}$ and a feasible transition sequence

$$(\emptyset, \emptyset, P) \xrightarrow{*} (\phi, \sigma, Q)$$

such that $\mathcal{A}(\phi, Q) = s$? **This problem is in NP**

Confidentiality

In the DY case

Is there a DY attacker $\mathcal{A}$ and a feasible transition sequence

$$(\emptyset, \emptyset, P) \xrightarrow{*} (\phi, \sigma, Q)$$

such that $\mathcal{A}(\phi, Q) = s$? **This problem is in NP**

In the computational case

Is there a PPT $\mathcal{A}$ such that, for every computational model $\mathcal{M}_c(\tau, \eta)$, the probability that there is a feasible sequence

$$(\emptyset, \emptyset, P) \xrightarrow{*} (\phi, \sigma, Q)$$

such that $\mathcal{A}(\phi, Q) = s$ is negligible in η ?
This requires in general assumptions on the libraries

Exercises

In the following cases, give reasonable processes A, B and either give an attack on the confidentiality of s or prove that there is no such attack in the DY model.

1.

$$\begin{array}{ll} A \to B: & \nu s, \nu r.\ \langle \mathsf{pk}(\mathsf{sk}_A), \mathsf{aenc}(s, \mathsf{pk}(\mathsf{sk}_B), r)\rangle \\ B \to A & \nu r'.\ \langle \mathsf{pk}(\mathsf{sk}_B), \mathsf{aenc}(s, \mathsf{pk}(\mathsf{sk}_A), r')\rangle \end{array}$$

$$P = \nu \mathsf{sk}_a, \nu \mathsf{sk}_b.\ \mathsf{out}(\langle \mathsf{pk}(\mathsf{sk}_A), \mathsf{pk}(\mathsf{sk}_B)\rangle) \cdot (A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_B)) \parallel B(\mathsf{sk}_b))$$

2.

$$\begin{array}{ll} A \to B: & \nu s, r_1, r_2.\ \mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_A), \mathsf{aenc}(s, \mathsf{pk}(\mathsf{sk}_B), r_1)\rangle, \mathsf{pk}(\mathsf{sk}_B), r_2) \\ B \to A: & \nu r_3, r_4.\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_B), \mathsf{aenc}(s, \mathsf{pk}(\mathsf{sk}_A), r_3)\rangle, \mathsf{pk}(\mathsf{sk}_A), r_4) \end{array}$$

$$P = \nu \mathsf{sk}_a, \nu \mathsf{sk}_b.\ \mathsf{out}(\langle \mathsf{pk}(\mathsf{sk}_A), \mathsf{pk}(\mathsf{sk}_B)\rangle) \cdot (A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_B)) \parallel B(\mathsf{sk}_b) \parallel B(\mathsf{sk}_b))$$

Gathering feasability conditions

States of the network are tuples $(\phi, \sigma, P, \theta)$, where

- ϕ, σ, P as before
- θ is a *constraint*: equalities, disequalities and computational constraints of the form $\phi \triangleright u$.

Gathering feasability conditions

States of the network are tuples $(\phi, \sigma, P, \theta)$, where

- ϕ, σ, P as before
- θ is a *constraint*: equalities, disequalities and computational constraints of the form $\phi \triangleright u$.

$$\frac{}{(\phi, \sigma, \mathsf{in}(x).P, \theta) \;\rightarrow\; (\phi, \sigma, P, \theta \wedge \phi \triangleright x)}$$

Gathering feasability conditions

States of the network are tuples $(\phi, \sigma, P, \theta)$, where

- ϕ, σ, P as before
- θ is a *constraint*: equalities, disequalities and computational constraints of the form $\phi \triangleright u$.

$$\frac{}{(\phi, \sigma, \mathsf{in}(x).P, \theta) \;\rightarrow\; (\phi, \sigma, P, \theta \wedge \phi \triangleright x)}$$

$$\frac{}{(\phi, \sigma, \text{if } \mathtt{EQ}(s,t) \text{ then } P \text{ else } Q, \theta) \;\rightarrow\; (\phi, \sigma, P, \theta \wedge \mathtt{EQ}(s,t))}$$

Gathering feasability conditions

States of the network are tuples $(\phi, \sigma, P, \theta)$, where

- ϕ, σ, P as before
- θ is a *constraint*: equalities, disequalities and computational constraints of the form $\phi \triangleright u$.

$$\frac{}{(\phi, \sigma, \mathsf{in}(x).P, \theta) \;\rightarrow\; (\phi, \sigma, P, \theta \wedge \phi \triangleright x)}$$

$$\frac{}{(\phi, \sigma, \text{if } \mathtt{EQ}(s,t) \text{ then } P \text{ else } Q, \theta) \;\rightarrow\; (\phi, \sigma, P, \theta \wedge \mathtt{EQ}(s,t))}$$

$$\frac{}{(\phi, \sigma, \text{if } \mathtt{EQ}(s,t) \text{ then } P \text{ else } Q, \theta) \;\rightarrow\; (\phi, \sigma, P, \theta \wedge \neg\mathtt{EQ}(s,t))}$$

Consequences

Advantages

- A finite transition system (regardless of the model)
- Confidentiality reduces to constraint satisfaction

$$\theta \wedge \phi_f \rhd s$$

in NP in the DY model

Consequences

Computational case

Specify the assumptions on the libraries: impossibility conditions.

$$\not\rhd n$$

$$S, \mathsf{aenc}(n, \mathsf{pk}(k), r) \rhd n \quad \Rightarrow \quad S \rhd n$$

$$S_1 \rhd x \wedge S_2, x \rhd y \quad \Rightarrow \quad S_1, S_2 \rhd y$$

$$S_1 \rhd x_1 \wedge \ldots \wedge S_n \rhd x_n \quad \Rightarrow \quad S_1, \ldots, S_n \rhd f(x_1, \ldots, x_n)$$

S, S_1, S_2 are finite sets of terms.

Consequences

Computational case

Specify the assumptions on the libraries: impossibility conditions.

$$\not\rhd n$$

$$S, \mathsf{aenc}(n, \mathsf{pk}(k), r) \rhd n \quad \Rightarrow \quad S \rhd n$$

$$S_1 \rhd x \wedge S_2, x \rhd y \quad \Rightarrow \quad S_1, S_2 \rhd y$$

$$S_1 \rhd x_1 \wedge \ldots \wedge S_n \rhd x_n \quad \Rightarrow \quad S_1, \ldots, S_n \rhd f(x_1, \ldots, x_n)$$

S, S_1, S_2 are finite sets of terms.

Check the constraint satisfiability, together with $\phi \rhd s$ and the above axioms (in PTIME !!)

Exercise

Back to the simple handshake protocol. Study its security in the computational model, assuming the properties of the cryptographic libraries that are described in the lecture.

Adding events to the process calculus

syntax

$Eva, Evb, \ldots$ is a set of event symbols.

$P ::=$		0	null process (stalled)
	$\mid$	$\text{in}(x).P$	input of x (binds x)
	$\mid$	$\text{out}(t).P$	output of t
	$\mid$	if $\texttt{EQ}(u, v)$ then P else P	conditional branching
	$\mid$	let $y = u$ in P	evaluation
	$\mid$	$\nu \overline{n}.P$	random generation
	$\mid$	$P \| P$	parallel composition
	$\mid$	$Eve(\overline{u}) \cdot P$	the event Eve is raised with a sequence of values $\overline{u}$

Adding events to the process calculus

semantics

The states have now an event component $\mathcal{E}$, a sequence of event symbols together with values.
$\mathcal{E}$ is only modified when an event occurs in the process:

$$\frac{}{(\phi, \sigma, \mathit{Eve}(\overline{u}) \cdot P, \mathcal{E}) \rightarrow (\phi, \sigma, P, \mathcal{E} \cdot \mathit{Eve}([\![\overline{u}]\!]_{\sigma}^{\mathcal{M}})}$$

Example

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) =$$
$$\nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r).$$
$$\mathsf{in}(z).\ \mathsf{let}\ z_1 = \mathsf{dec}(z, \mathsf{sk}_a)\ \mathsf{in}$$
$$\mathsf{if}\ \mathtt{EQ}(z_1, n)\ \mathsf{then}\ \mathit{Eva}(n, \mathsf{pk}(\mathsf{sk}_b), \mathsf{pk}(\mathsf{sk}_a))\ \mathsf{else}\ \mathbf{0}$$

$$B(\mathsf{sk}_b) =$$
$$\nu r'.\ \mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}$$
$$\mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}$$
$$\mathsf{out}(\mathsf{aenc}(y_2, y_1, r')).\mathit{Evb}(y_2, \mathsf{pk}(\mathsf{sk}_b), y_1)\ \mathbf{0}.$$

Example

$$A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) = \nu n, r.\ \mathsf{out}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_a), n\rangle, \mathsf{pk}(\mathsf{sk}_b), r).\ \mathsf{in}(z).\ \mathsf{let}\ z_1 = \mathsf{dec}(z, \mathsf{sk}_a)\ \mathsf{in}\ \mathsf{if}\ \mathtt{EQ}(z_1, n)\ \mathsf{then}\ \mathit{Eva}(n, \mathsf{pk}(\mathsf{sk}_b), \mathsf{pk}(\mathsf{sk}_a))\ \mathsf{else}\ \mathbf{0}$$

$$B(\mathsf{sk}_b) = \nu r'.\ \mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r')).\mathit{Evb}(y_2, \mathsf{pk}(\mathsf{sk}_b), y_1)\ \mathbf{0}.$$

$$\forall x, y, z.\quad \mathit{Eva}(x, y, z) \Rightarrow \mathit{Evb}(x, y, z)$$

Agreement property

$$\forall x, y, z. \quad Eva(x, y, z) \Rightarrow Evb(x, y, z)$$

More generally

$$\forall \vec{x}. \quad Eva_1(\overline{u_1}), \ldots, Eva_k(\overline{u_k}) \Rightarrow Eva(\overline{u})$$

For every feasible trace, ending with an event set $\mathcal{E}$, for any assignment σ, if $Eva_1(\overline{u_1}\sigma), \ldots, Eva_k(\overline{u_k}\sigma) \in \mathcal{E}$, then $Eva(\overline{u}\sigma) \in \mathcal{E}$

Exercise

In the DY-mode, does the simple handshake protocol satisfy the agreement property ?
What happens if

1. we move forward the *Eva*
2. *B* replies with y_2 (instead of the encrypted version) ?

Adding replication in the calculus

$P ::=$		0	null process
	$\mid$	$\mathsf{in}(x).P$	input of x (binds x)
	$\mid$	$\mathsf{out}(t).P$	output of t
	$\mid$	if $\mathtt{EQ}(u, v)$ then P else P	conditional branching
	$\mid$	let $y = u$ in P	evaluation
	$\mid$	$\nu \overline{n}.P$	random generation
	$\mid$	$P \Vert P$	parallel composition
	$\mid$	$Eve(\overline{u}) \cdot P$	the event Eve is raised with a sequence of values $\overline{u}$
	$\mid$	$!P$	

Adding replication in the calculus

$P ::=$	0	null process
$\mid$	$\mathrm{in}(x).P$	input of x (binds x)
$\mid$	$\mathrm{out}(t).P$	output of t
$\mid$	if $\mathtt{EQ}(u, v)$ then P else P	conditional branching
$\mid$	let $y = u$ in P	evaluation
$\mid$	$\nu \overline{n}.P$	random generation
$\mid$	$P \Vert P$	parallel composition
$\mid$	$Eve(\overline{u}) \cdot P$	the event Eve is raised with a sequence of values $\overline{u}$
$\mid$	$!P$	

Operational semantics:

$$!P \quad \equiv \quad P \Vert !P$$

Example

The handshake protocol

$$!(\nu \mathsf{sk}_a, \mathsf{sk}_b.\quad \mathsf{out}(\langle \mathsf{pk}(\mathsf{sk}_a), \mathsf{pk}(\mathsf{sk}_b)\rangle).\quad (!A(\mathsf{sk}_a, \mathsf{pk}(\mathsf{sk}_b)) \parallel !B(\mathsf{sk}_b))$$

Verification in the DY-model

- This is undecidable
- Most popular: approximation and translation into Horn clauses

Translation into Horn clauses

Attacker clauses

$$
\begin{aligned}
\mathtt{Att}(a) &\Leftarrow \\
\mathtt{Att}(\langle x_1, x_2 \rangle) &\Leftarrow \mathtt{Att}(x_1),\ \mathtt{Att}(x_2) \\
\mathtt{Att}(\mathsf{pk}(x)) &\Leftarrow \mathtt{Att}(x) \\
\mathtt{Att}(\mathsf{aenc}(x_1, x_2, x_3)) &\Leftarrow \mathtt{Att}(x_1)\ \mathtt{Att}(x_2),\ \mathtt{Att}(x_3) \\
\mathtt{Att}(x_1) &\Leftarrow \mathtt{Att}(\mathsf{aenc}(x_1, \mathsf{pk}(x_2), x_3)),\ \mathtt{Att}(x_2) \\
\mathtt{Att}(x_1) &\Leftarrow \mathtt{Att}(\langle x_1, x_2 \rangle) \\
\mathtt{Att}(x_2) &\Leftarrow \mathtt{Att}(\langle x_1, x_2 \rangle)
\end{aligned}
$$

a is a free name.

Translation into Horn clauses

Protocol clauses

$T(P) = [\![P]\!]_{\emptyset,\emptyset}$ where $[\![P]\!]_{\rho,H}$ is defined as follows

$$
\begin{array}{rcl}
[\![\mathbf{0}]\!] & = & \emptyset \\
[\![P \| Q]\!]_{\rho,H} & = & [\![P]\!]_{\rho,H} \cup [\![Q]\!]_{\rho,H} \\
[\![!P]\!]_{\rho,H} & = & [\![P]\!]_{\rho,H} \\
[\![\nu n.\ P]\!]_{\rho,H} & = & [\![P]\!]_{\rho \uplus \{n \mapsto n(H)\},H} \\
[\![\mathsf{in}(x).P]\!]_{\rho,H} & = & [\![P]\!]_{\rho,H \uplus \{x\}} \\
[\![\mathsf{out}(s).P]\!]_{\rho,H} & = & [\![P]\!]_{\rho,H} \ \cup \ \{\mathtt{Att}(s\rho) \Leftarrow \mathtt{Att}(s_1\rho), \ldots \mathtt{A} \\
[\![\mathsf{if}\ \mathtt{EQ}(s,t)\ \mathsf{then}\ P\ \mathsf{else}\ Q]\!]_{\rho,H} & = & [\![P]\!]_{\rho\sigma,H\sigma} \cup [\![Q]\!]_{\rho,H} \\
[\![\mathsf{let}\ x = s\ \mathsf{in}\ P]\!]_{\rho,H} & = & [\![P]\!]_{\rho\sigma_{x,s},H\sigma_{x,s}} \\
[\![\mathsf{let}\ x = s\ \mathsf{in}\ P]\!]_{\rho,H} & = & \emptyset
\end{array}
$$

Examples

$B = \nu r.\mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r))$

Examples

$B = \nu r.\mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r))$

$x = \mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)$:

$B = \nu r.\mathsf{in}(\mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)).\mathsf{out}(\mathsf{aenc}(y_2, y_1, r))$

Examples

$B = \nu r.\mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r))$

$x = \mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)$:

$$B = \nu r.\mathsf{in}(\mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)).\mathsf{out}(\mathsf{aenc}(y_2, y_1, r))$$

$$\mathtt{Att}(\mathsf{aenc}(y_2, y_1, r(\cdots))) \quad \Leftarrow \quad \mathtt{Att}(\mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)))$$

Examples

$B = \nu r.\mathsf{in}(x).\mathsf{let}\ y = \mathsf{dec}(x, \mathsf{sk}_b)\ \mathsf{in}\ \mathsf{let}\ y_1 = \pi_1(y)\ \mathsf{in}\ \mathsf{let}\ y_2 = \pi_2(y)\ \mathsf{in}\ \mathsf{out}(\mathsf{aenc}(y_2, y_1, r))$

$x = \mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)$:

$$B = \nu r.\mathsf{in}(\mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)).\mathsf{out}(\mathsf{aenc}(y_2, y_1, r))$$

$$\mathtt{Att}(\mathsf{aenc}(y_2, y_1, r(\cdots))) \quad \Leftarrow \quad \mathtt{Att}(\mathsf{aenc}(\langle y_1, y_2 \rangle, \mathsf{pk}(\mathsf{sk}_B), z)))$$

$$\mathtt{Att}(\mathsf{aenc}(\langle \mathsf{pk}(\mathsf{sk}_A), n \rangle, \mathsf{pk}(\mathsf{sk}_B), r')) \quad \Leftarrow$$

ProVerif

A tool available online, whose main developer is Bruno Blanchet.

- Translate the specification of primitives into rewriting system + Attacker clauses
- Translate the protocol into clauses
- Negate the security goal
- Check for satisfiability (Ordered resolution strategy).
  Three possible outputs:
  1. Unsatisfiable: the protocol is secure
  2. Satisfiable: there might be an attack (but might also be secure)
  3. Non termination

Very successful tool : works well in practice.

False attacks

Using twice the same protocol clause (rigid variables !)

$$
\begin{array}{lll}
A \to B: & \nu n_1, n_2, r_1, r_2. & \langle \mathsf{aenc}(n_1, \mathsf{pk}(\mathsf{sk}_B), r_1), \mathsf{aenc}(n_2, \mathsf{pk}(\mathsf{sk}_B), r_2) \rangle \\
B \to A: & \nu n, r_3. & \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r_3) \\
A \to B: & \nu s. & \langle n, \mathsf{senc}(s, \langle n_1, n_2 \rangle) \rangle
\end{array}
$$

False attacks

Using twice the same protocol clause (rigid variables !)

$$
\begin{array}{lll}
A \to B: & \nu n_1, n_2, r_1, r_2. & \langle \mathsf{aenc}(n_1, \mathsf{pk}(\mathsf{sk}_B), r_1), \mathsf{aenc}(n_2, \mathsf{pk}(\mathsf{sk}_B), r_2) \rangle \\
B \to A: & \nu n, r_3. & \mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_A), r_3) \\
A \to B: & \nu s. & \langle n, \mathsf{senc}(s, \langle n_1, n_2 \rangle) \rangle
\end{array}
$$

Approximating names

$$
\begin{array}{ll}
A \to B: & \nu n.\mathsf{aenc}(n, \mathsf{pk}(\mathsf{sk}_B), r) \\
B \to A: & \nu n'.\mathsf{aenc}(n', \mathsf{pk}(\mathsf{sk}_A), r') \\
A \to B: & \nu s.\mathsf{in}(\mathsf{aenc}(x, \mathsf{pk}(\mathsf{sk}_A), y).\mathsf{if}\ x = n\ \mathsf{then}\ \mathsf{out}(s)\ \mathsf{else}\ \mathsf{out}(n)
\end{array}
$$

The determinacy issue

Currently the calculus is non-deterministic, while the computational security is probabilistic.
As security is an asymptotic propery, it is harmless when the (symbolic) transition sytem is finite (for a fixed attacker).

The determinacy issue

Currently the calculus is non-deterministic, while the computational security is probabilistic.
As security is an asymptotic propery, it is harmless when the (symbolic) transition sytem is finite (for a fixed attacker).

Instead of non-determinism, we let the attacker choose the transition.

The determinacy issue

Currently the calculus is non-deterministic, while the computational security is probabilistic.
As security is an asymptotic propery, it is harmless when the (symbolic) transition sytem is finite (for a fixed attacker).

Instead of non-determinism, we let the attacker choose the transition.

Communications are using different *communication channels*, that solve the non-determinism and are chosen by the attacker.

Extending the calculus with channels

$c, c_1, \ldots$ are channel names

$P ::=$	0	null process (stalled)
$\mid$	$\mathrm{in}(c, x).P$	input of x (binds x)
$\mid$	$\mathrm{out}(c, t).P$	output of t
$\mid$	$\mathrm{if}\ \mathtt{EQ}(u, v)\ \mathrm{then}\ P\ \mathrm{else}\ P$	conditional branching
$\mid$	$\mathrm{let}\ y = u\ \mathrm{in}\ P$	evaluation
$\mid$	$\nu \overline{n}.P$	random generation
$\mid$	$P \Vert P$	parallel composition
$\mid$	$Eve(\overline{u}) \cdot P$	the event Eve is raised with a sequence of values $\overline{u}$
$\mid$	$!P$	
$\mid$	$\nu c.P$	

Operational semantics

Only known channels can be observed

$$\frac{}{(\phi, \sigma, \mathsf{out}(c, u) \cdot P, \mathcal{E}, \theta) \rightarrow (\phi \cdot [\![u]\!]^{\mathcal{M}}_{\sigma}, \sigma, P, \mathcal{E}, \theta \wedge \phi \rhd c)}$$

Operational semantics

Only known channels can be observed

$$\frac{}{(\phi, \sigma, \mathsf{out}(c, u) \cdot P, \mathcal{E}, \theta) \;\rightarrow\; (\phi \cdot [\![u]\!]^{\mathcal{M}}_{\sigma}, \sigma, P, \mathcal{E}, \theta \wedge \phi \triangleright c)}$$

Inputs are only possible on the designated channels

$$\frac{}{(\phi, \sigma, \mathsf{in}(c, x) \cdot P, \mathcal{E}, \theta) \;\rightarrow\; (\phi, \sigma \uplus \{x \mapsto [\![u]\!]^{\mathcal{M}}_{\sigma}\}, P, \mathcal{E}, \theta)}$$

If $\mathcal{A}(\phi, \gamma) = [\![c]\!]^{\mathcal{M}}$ (γ is the global control state).

Internal synchronizations

$$(\phi, \sigma, \mathsf{in}(c, x) \cdot P \| \mathsf{out}(c, u) \cdot Q, \mathcal{E}, \theta) \rightarrow (\phi, \sigma \uplus \{x \mapsto [\![u]\!]_\sigma^{\mathcal{M}}\}, P \| Q, \mathcal{E}, \theta)$$

Restrictions on the processes

We assume that, for any reachable process

$$\mathsf{in}(c_1, x_1).P_1 \parallel \cdots \parallel \mathsf{in}(c_k, x_k).P_k \parallel Q$$

$c_1, \ldots, c_k$ are pairwise distinct.

Restrictions on the processes

We assume that, for any reachable process

$$\mathsf{in}(c_1, x_1).P_1 \parallel \cdots \parallel \mathsf{in}(c_k, x_k).P_k \parallel Q$$

$c_1, \ldots, c_k$ are pairwise distinct.

Lemma

Any process P without channel can be translated into a process P' such that P' satisfies the above restriction and any P has the same (DY) operational semantics as forgetting the channels in the semantics of P'.

Example

$$P = \mathsf{in}(x).\mathsf{out}(t)$$

$$Q = \mathsf{in}(y).\mathsf{out}(u)$$

$$\mathsf{Trans}(P \| Q) = \mathsf{in}(c_P, x).\mathsf{out}(c_P, t) \parallel \mathsf{in}(c_Q, y).\mathsf{out}(c_Q, u)$$

Example

$$P = \mathsf{in}(x).\mathsf{out}(t)$$

$$Q = \mathsf{in}(y).\mathsf{out}(u)$$

$$\mathsf{Trans}(P \| Q) = \mathsf{in}(c_P, x).\mathsf{out}(c_P, t) \parallel \mathsf{in}(c_Q, y).\mathsf{out}(c_Q, u)$$

$$\mathsf{Trans}(!(P \| Q)) = \begin{array}{l} ! \ \mathsf{in}(c_{!(P\|Q)}, ?).\nu c_P, c_Q.\mathsf{out}(c_{!(P\|Q)}, \langle c_P, c_Q \rangle). \\ \mathsf{Trans}(P \| Q) \end{array}$$

Indistinguishability

$$\mathsf{Tr}(\mathcal{M}, \mathcal{A}, P) = m_1, \cdots, m_k, \cdots$$

the (unique) sequence of outputs of P, with attacker $\mathcal{A}$ in model $\mathcal{M}$.

Indistinguishability

$$\mathsf{Tr}(\mathcal{M}, \mathcal{A}, P) = m_1, \cdots, m_k, \cdots$$

the (unique) sequence of outputs of P, with attacker $\mathcal{A}$ in model $\mathcal{M}$.

Informally: P_1 is *indistinguishable* from P_2, for a (familly of) model(s) $\mathcal{M}$, if, for every $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$,

$$\mathcal{A}_2(\mathsf{Tr}(\mathcal{M}, \mathcal{A}_1, P_1)) \;=\; \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}, \mathcal{A}_1, P_1))$$

iff

$$\mathcal{A}_2(\mathsf{Tr}(\mathcal{M}, \mathcal{A}_1, P_2)) \;=\; \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}, \mathcal{A}_1, P_2))$$

Examples in the DY case (I)

$\mathcal{A}_1$ gives (deducible) inputs to the process and $\mathcal{A}_2, \mathcal{A}_3$ observe some identities on the output.

Examples in the DY case (I)

$\mathcal{A}_1$ gives (deducible) inputs to the process and $\mathcal{A}_2, \mathcal{A}_3$ observe some identities on the output.

Example 1

$$P_1 = \nu n.\mathrm{out}(c, n) \qquad P_2 = \nu n'.\mathrm{out}(c, n')$$

Examples in the DY case (I)

$\mathcal{A}_1$ gives (deducible) inputs to the process and $\mathcal{A}_2, \mathcal{A}_3$ observe some identities on the output.

Example 1

$$P_1 = \nu n.\text{out}(c, n) \qquad P_2 = \nu n'.\text{out}(c, n')$$

$P_1 \sim P_2$
$\mathcal{A}_1$ can't input anything and $\mathcal{A}_2(n) = \mathcal{A}_3(n)$ iff $\mathcal{A}_2 = \mathcal{A}_3$

Examples in the DY case (I)

$\mathcal{A}_1$ gives (deducible) inputs to the process and $\mathcal{A}_2, \mathcal{A}_3$ observe some identities on the output.

Example 1

$$P_1 = \nu n.\mathsf{out}(c, n) \qquad P_2 = \nu n'.\mathsf{out}(c, n')$$

$P_1 \sim P_2$
$\mathcal{A}_1$ can't input anything and $\mathcal{A}_2(n) = \mathcal{A}_3(n)$ iff $\mathcal{A}_2 = \mathcal{A}_3$

Example 2

$$P_1 = \mathsf{out}(c, \mathtt{ok}) \qquad P_2 = \nu n.\mathsf{out}(c, n)$$

Examples in the DY case (I)

$\mathcal{A}_1$ gives (deducible) inputs to the process and $\mathcal{A}_2, \mathcal{A}_3$ observe some identities on the output.

Example 1

$$P_1 = \nu n.\mathsf{out}(c, n) \qquad P_2 = \nu n'.\mathsf{out}(c, n')$$

$P_1 \sim P_2$
$\mathcal{A}_1$ can't input anything and $\mathcal{A}_2(n) = \mathcal{A}_3(n)$ iff $\mathcal{A}_2 = \mathcal{A}_3$

Example 2

$$P_1 = \mathsf{out}(c, \mathtt{ok}) \qquad P_2 = \nu n.\mathsf{out}(c, n)$$

$P_1 \not\sim P_2$
$\mathcal{A}_2$ is the identity, $\mathcal{A}_3$ computes $\mathtt{ok}$.

Examples in the DY case (II)

Example 3

$$P_1 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_1, \mathsf{pk}(k), r))$$

$$P_2 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_2, \mathsf{pk}(k), r))$$

Examples in the DY case (II)

Example 3

$$P_1 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_1, \mathsf{pk}(k), r))$$

$$P_2 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_2, \mathsf{pk}(k), r))$$

$P_1 \not\sim P_2$

Examples in the DY case (II)

Example 3

$$P_1 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_1, \mathsf{pk}(k), r))$$

$$P_2 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_2, \mathsf{pk}(k), r))$$

$P_1 \not\sim P_2$

Example 4

$$P_1 = \mathsf{in}(c, x) \cdot \mathsf{out}(c, \langle x, n \rangle)$$

$$P_2 = \mathsf{in}(c, x) \cdot \mathsf{out}(c, \langle n, x \rangle)$$

Examples in the DY case (II)

Example 3

$$P_1 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_1, \mathsf{pk}(k), r))$$

$$P_2 = \nu k.\mathsf{out}(\mathsf{pk}(k)).\mathsf{out}(c, \mathsf{aenc}(n_2, \mathsf{pk}(k), r))$$

$P_1 \not\sim P_2$

Example 4

$$P_1 = \mathsf{in}(c, x) \cdot \mathsf{out}(c, \langle x, n \rangle)$$

$$P_2 = \mathsf{in}(c, x) \cdot \mathsf{out}(c, \langle n, x \rangle)$$

$P_1 \not\sim P_2$

Results in the DY case

Indistinguishability is decidable in the DY case (for processes without replication). The complexity is unknown.

Verification tool: APTE.

An approximated equivalence is considered in PROVERIF (works with replication)

Computational indistinguishability (I)

For any PPT $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$,

$$\begin{aligned}&\mathbf{Prob}\{\tau, \rho : \mathcal{A}_2(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_1)) = \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_1))\} \\ -&\mathbf{Prob}\{\tau, \rho : \mathcal{A}_2(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_2)) = \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_2))\}\end{aligned}$$

is negligible.

Computational indistinguishability (I)

For any PPT $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$,

$$\begin{aligned}&\mathbf{Prob}\{\tau, \rho : \mathcal{A}_2(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_1)) = \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_1))\} \\ -&\mathbf{Prob}\{\tau, \rho : \mathcal{A}_2(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_2)) = \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_2))\}\end{aligned}$$

is negligible.

Example 1

$$P_1 = \nu n.\mathsf{out}(c, n) \qquad P_2 = \nu n'.\mathsf{out}(c, n')$$

Computational indistinguishability (I)

For any PPT $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$,

$$\begin{aligned}&\textbf{Prob}\{\tau, \rho : \mathcal{A}_2(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_1)) = \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_1))\} \\ -&\textbf{Prob}\{\tau, \rho : \mathcal{A}_2(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_2)) = \mathcal{A}_3(\mathsf{Tr}(\mathcal{M}_c(\tau, \eta), \mathcal{A}_1, P_2))\}\end{aligned}$$

is negligible.

Example 1

$$P_1 = \nu n.\mathsf{out}(c, n) \qquad P_2 = \nu n'.\mathsf{out}(c, n')$$

$P_1 \sim P_2$

The difference of probabilities is 0 !!

Computational indistinguishability (II)

Example 2

$$P_1 = \nu k_1, r_1, r_2.\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_1), r_1)).\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_1), r_2))$$

$$P_2 = \nu k_1, k_2, r_1, r_2.\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_1), r_1)).\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_2), r_2))$$

Computational indistinguishability (II)

Example 2

$P_1 = \nu k_1, r_1, r_2.\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_1), r_1)).\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_1), r_2))$

$P_2 = \nu k_1, k_2, r_1, r_2.\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_1), r_1)).\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_2), r_2))$

$P_1 \sim P_2$ depends on assumptions on the crypto libraries

Computational indistinguishability (II)

Example 2

$$P_1 = \nu k_1, r_1, r_2.\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_1), r_1)).\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_1), r_2))$$

$$P_2 = \nu k_1, k_2, r_1, r_2.\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_1), r_1)).\mathsf{out}(c, \mathsf{aenc}(\mathtt{ok}, \mathsf{pk}(k_2), r_2))$$

$P_1 \sim P_2$ depends on assumptions on the crypto libraries

Example 3

$$P_1 = \mathsf{in}(c, x).\mathsf{if}\ x = 0\ \mathsf{then}\ \mathsf{out}(c, 0)\ \mathsf{else}\ \mathsf{out}(c, x)$$

$$P_2 = \mathsf{in}(c, x).\mathsf{out}(c, x)$$

Computational indistinguishability (II)

Example 2

$P_1 = \nu k_1, r_1, r_2.\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_1), r_1)).\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_1), r_2))$

$P_2 = \nu k_1, k_2, r_1, r_2.\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_1), r_1)).\text{out}(c, \text{aenc}(\text{ok}, \text{pk}(k_2), r_2))$

$P_1 \sim P_2$ depends on assumptions on the crypto libraries

Example 3

$$P_1 = \text{in}(c, x).\text{if } x = 0 \text{ then out}(c, 0) \text{ else out}(c, x)$$

$$P_2 = \text{in}(c, x).\text{out}(c, x)$$

$P_1 \sim P_2$

Attacker's inputs

Example 4

$$P_1 = \nu n_1, n_2.\mathsf{out}(\langle n_1, n_2 \rangle).\mathsf{in}(x).\mathsf{if}\ \pi_1(x) = n_1\ \mathsf{then}\ \mathsf{out}(n_2)\ \mathsf{else}\ \mathbf{0}$$

$P_2 = P_1$

We cannot use the constraints $\phi \triangleright x$ any more.

Attacker's inputs

Example 4

$$P_1 = \nu n_1, n_2.\mathsf{out}(\langle n_1, n_2\rangle).\mathsf{in}(x).\mathsf{if}\ \pi_1(x) = n_1\ \mathsf{then}\ \mathsf{out}(n_2)\ \mathsf{else}\ \mathbf{0}$$

$P_2 = P_1$

We cannot use the constraints $\phi \triangleright x$ any more.

We use fresh function symbols $g_1, g_2 \cdots$ for successive attacker's inputs.

$$P_1 = \nu n_1, n_2.\mathsf{out}(\langle n_1, n_2\rangle).\mathsf{if}\ \pi_1(g_1(\langle n_1, n_2\rangle)) = n_1\ \mathsf{then}\ \mathsf{out}(n_2)\ \mathsf{else}\ \mathbf{0}$$

Attacker's inputs

Examples

$$P_1 = \quad \mathsf{out}(u).\mathsf{in}(x).\mathsf{out}(v).\\ \quad\quad\quad\quad \text{if } b \text{ then } \mathsf{in}(y).Q_1 \text{ else } \mathsf{in}(z).Q_2$$

$$P_2 = \quad \mathsf{out}(u').\mathsf{in}(x').\mathsf{out}(v').\\ \quad\quad\quad\quad \mathsf{in}(y') \cdot Q'$$

Attacker's inputs

Examples

$$P_1 = \quad \mathsf{out}(u).\mathsf{in}(x).\mathsf{out}(v).\\ \quad\quad\quad\quad \text{if } b \text{ then } \mathsf{in}(y).Q_1 \text{ else } \mathsf{in}(z).Q_2$$

$$P_2 = \quad \mathsf{out}(u').\mathsf{in}(x').\mathsf{out}(v').\\ \quad\quad\quad\quad \mathsf{in}(y') \cdot Q'$$

$$P_1 = \quad \mathsf{out}(u).\mathsf{out}(v\{x \mapsto g_1(u)\})).\\ \quad\quad\quad\quad \text{if } b \text{ then } Q_1\{y \mapsto g_2(u,w) \text{ else } Q_2\{z \mapsto g_2(u,w)\}$$

where $w = v\{x \mapsto g_1(u)\}$

Attacker's inputs

Examples

$$P_1 = \quad \mathsf{out}(u).\mathsf{in}(x).\mathsf{out}(v).\\ \mathsf{if}\ b\ \mathsf{then}\ \mathsf{in}(y).Q_1\ \mathsf{else}\ \mathsf{in}(z).Q_2$$

$$P_2 = \quad \mathsf{out}(u').\mathsf{in}(x').\mathsf{out}(v').\\ \mathsf{in}(y') \cdot Q'$$

$$P_1 = \quad \mathsf{out}(u).\mathsf{out}(v\{x \mapsto g_1(u)\})).\\ \mathsf{if}\ b\ \mathsf{then}\ Q_1\{y \mapsto g_2(u,w)\ \mathsf{else}\ Q_2\{z \mapsto g_2(u,w)\}$$

where $w = v\{x \mapsto g_1(u)\}$

$$P_2 = \quad \mathsf{out}(u').\mathsf{out}(v'\{x' \mapsto g_1(u')\}).\\ Q'\{y' \mapsto g_2(u',w')\}$$

where $w' = v'\{x' \mapsto g_1(u')\}$

The folding trick

Example

$$P_1 = \mathsf{in}(x).\mathsf{if}\ \mathtt{EQ}(x, 0)\ \mathsf{then}\ \mathsf{out}(c, 0)\ \mathsf{else}\ \mathsf{out}(c, \langle x, 0\rangle)$$

$$t_{P_1} = \mathsf{if}\ \mathtt{EQ}(g_1(), 0)\ \mathsf{then}\ 0\ \mathsf{else}\ \langle g_1(), 0\rangle$$

if _ then _ else _ is now a function symbol !

The folding trick

Example

$$P_1 = \mathsf{in}(x).\mathsf{if}\ \mathtt{EQ}(x, 0)\ \mathsf{then}\ \mathsf{out}(c, 0)\ \mathsf{else}\ \mathsf{out}(c, \langle x, 0\rangle)$$

$$t_{P_1} = \mathsf{if}\ \mathtt{EQ}(g_1(), 0)\ \mathsf{then}\ 0\ \mathsf{else}\ \langle g_1(), 0\rangle$$

if _ then _ else _ is now a function symbol !

Example

$$P = \mathsf{in}(c, x) \cdot \mathsf{out}(c, u) \| \mathsf{in}(c', y) \cdot \mathsf{out}(c', v)$$

$$t_P = \mathsf{if}\ to() = c\ \mathsf{then}\ u\{x \mapsto g_1()\}\ \mathsf{else\ if}\ to() = c'\ \mathsf{then}\ \cdots\ \mathsf{else}\ \mathbf{0}$$

The folding trick

Example

$$P_1 = \mathsf{in}(x).\mathsf{if}\ \mathtt{EQ}(x, 0)\ \mathsf{then}\ \mathsf{out}(c, 0)\ \mathsf{else}\ \mathsf{out}(c, \langle x, 0\rangle)$$

$$t_{P_1} = \mathsf{if}\ \mathtt{EQ}(g_1(), 0)\ \mathsf{then}\ 0\ \mathsf{else}\ \langle g_1(), 0\rangle$$

if _ then _ else _ is now a function symbol !

Example

$$P = \mathsf{in}(c, x) \cdot \mathsf{out}(c, u) \| \mathsf{in}(c', y) \cdot \mathsf{out}(c', v)$$

$$t_P = \mathsf{if}\ to() = c\ \mathsf{then}\ u\{x \mapsto g_1()\}\ \mathsf{else\ if}\ to() = c'\ \mathsf{then}\ \cdots\ \mathsf{else}\ \mathbf{0}$$

In general, t_P is a sequence of terms using additional functions $g_1, g_2, ..., to, ...$ and if, EQ

Examples of axioms

Some axioms independent of the libraries

$$\begin{array}{rcl} \text{if } x \text{ then } y \text{ else } y & \sim & y \\ \text{if } x \text{ then if } x \text{ then } y \text{ else } y' \text{ else } z & \sim & \text{if } x \text{ then } y \text{ else } z \\ \text{if } \mathtt{EQ}(u,v) \text{ then } C[u] \text{ else } w & \sim & \text{if } \mathtt{EQ}(u,v) \text{ then } C[v] \text{ else } w \end{array}$$

$$x_1 \sim y_1 \wedge \cdots \wedge x_n \sim y_n \quad \Rightarrow \quad f(x_1, \ldots, x_n) \sim f(y_1, \ldots, y_n)$$

An IND-CPA axiom (assumption on the library)

$$\overline{u}, \mathsf{aenc}(x, \mathsf{pk}(k), r) \sim \overline{u}, \mathsf{aenc}(y, \mathsf{pk}(k), r)$$

if k, does not occur (as plaintext) in $\overline{u}, x, y$, r does not occur in $\overline{u}, x, y$.

Reducing the existence of an attack to satisfiability

P_1 is indistinguishable from P_2 iff

$$t_{P_1} \not\sim t_{P_2} \quad \wedge \text{Axioms}$$

is unsatisfiable. (The symbols $g_1, g_2, \cdots$ are part of the interpretation).

Reducing the existence of an attack to satisfiability

P_1 is indistinguishable from P_2 iff

$$t_{P_1} \not\sim t_{P_2} \quad \wedge \text{Axioms}$$

is unsatisfiable. (The symbols $g_1, g_2, \cdots$ are part of the interpretation).

Is it decidable (for processes without replication) ?

The main ideas

- Several possible interpretation domains/attacker models
- We cannot use the model-checking tools
- Tricky security properties
- The abstract symbolic semantics
- Axiomatizing what is *not possible*: Security is FO inconsistency