Verification of security protocols from confidentiality to privacy

Stéphanie Delaune

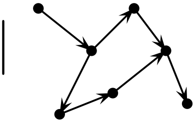
LSV, CNRS & ENS Cachan, France

Wednesday, August 26th, 2015

This talk: formal methods for protocol verification

Does the protocol *satisfy* a security property?

Modelling

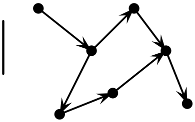


$\models \varphi$

This talk: formal methods for protocol verification

Does the protocol *satisfy* a security property?

Modelling



$\models \varphi$

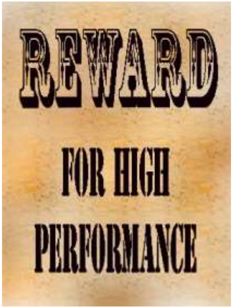
Two main tasks

1. Modelling cryptographic protocols and their security properties
2. Designing verification algorithms

Challenge

Would you be able to find the attack on the well-known Needham-Schroeder protocol?

$$
\begin{array}{ll}
A \rightarrow B: & \{A, N_a\}_{\mathsf{pub}(B)} \\
B \rightarrow A: & \{N_a, N_b\}_{\mathsf{pub}(A)} \\
A \rightarrow B: & \{N_b\}_{\mathsf{pub}(B)}
\end{array}
$$



To help you:

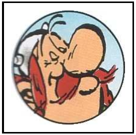
`http://www.lsv.ens-cachan.fr/~delaune/VTSA/proverif.pdf`

Needham-Schroeder's Protocol (1978)

- $$\begin{array}{lll} A & \to B: & \{A, N_a\}_{\mathrm{pub}(B)} \\ B & \to A: & \{N_a, N_b\}_{\mathrm{pub(A)}} \\ A & \to B: & \{N_b\}_{\mathrm{pub}(B)} \end{array}$$

Needham-Schroeder's Protocol (1978)

- $$\begin{array}{lll} A & \to B: & \{A, N_a\}_{\mathsf{pub}(B)} \\ B & \to A: & \{N_a, N_b\}_{\mathsf{pub(A)}} \\ A & \to B: & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

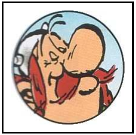


Needham-Schroeder's Protocol (1978)

$$\begin{array}{llll} & A & \to B : & \{A, N_a\}_{\mathrm{pub}(B)} \\ & B & \to A : & \{N_a, N_b\}_{\mathrm{pub(A)}} \\ \bullet & A & \to B : & \{N_b\}_{\mathrm{pub}(B)} \end{array}$$

Needham-Schroeder's Protocol (1978)

$$\begin{array}{lll} A & \rightarrow B: & \{A, N_a\}_{\mathrm{pub}(B)} \\ B & \rightarrow A: & \{N_a, N_b\}_{\mathrm{pub(A)}} \\ A & \rightarrow B: & \{N_b\}_{\mathrm{pub}(B)} \end{array}$$



Needham-Schroeder's Protocol (1978)

$$\begin{array}{lll} A & \to B: & \{A, N_a\}_{\mathrm{pub}(B)} \\ B & \to A: & \{N_a, N_b\}_{\mathrm{pub(A)}} \\ A & \to B: & \{N_b\}_{\mathrm{pub}(B)} \end{array}$$

Questions

- Is N_b secret between A and B ?
- When B receives $\{N_b\}_{\mathrm{pub}(B)}$, does this message really comes from A ?

Needham-Schroeder's Protocol (1978)

$$\begin{array}{lll} A & \rightarrow B: & \{A, N_a\}_{\mathsf{pub}(B)} \\ B & \rightarrow A: & \{N_a, N_b\}_{\mathsf{pub(A)}} \\ A & \rightarrow B: & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

Questions

- Is N_b secret between A and B ?
- When B receives $\{N_b\}_{\mathsf{pub}(B)}$, does this message really comes from A ?

Attack

An attack was found 17 years after its publication! [Lowe 96]

Man in the middle attack

Agent A

Attacker C

Agent B

Attack

- involving 2 sessions in parallel,
- an honest agent has to initiate a session with C.

$$
\begin{array}{lll}
A \rightarrow B & : & \{A, N_a\}_{\mathrm{pub}(B)} \\
B \rightarrow A & : & \{N_a, N_b\}_{\mathrm{pub}(A)} \\
A \rightarrow B & : & \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$

Man in the middle attack

$\xrightarrow{\{A, N_a\}_{\mathrm{pub}(C)}}$ $\xrightarrow{\{A, N_a\}_{\mathrm{pub}(B)}}$

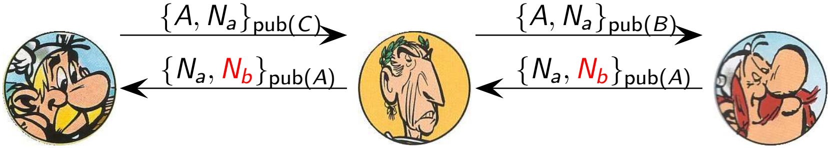
Agent A

Attacker C

Agent B

$$
\begin{array}{lcl}
\mathrm{A} \rightarrow \mathrm{B} & : & \{A, N_a\}_{\mathrm{pub}(B)} \\
\mathrm{B} \rightarrow \mathrm{A} & : & \{N_a, N_b\}_{\mathrm{pub}(A)} \\
\mathrm{A} \rightarrow \mathrm{B} & : & \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$

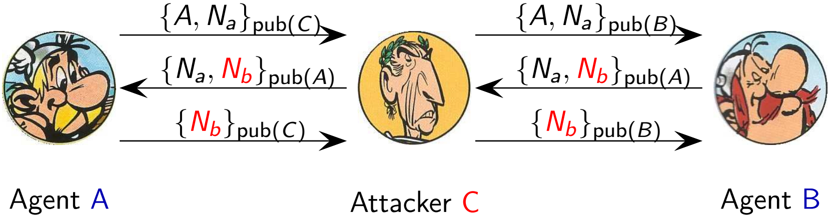
Man in the middle attack



Agent A | Attacker C | Agent B

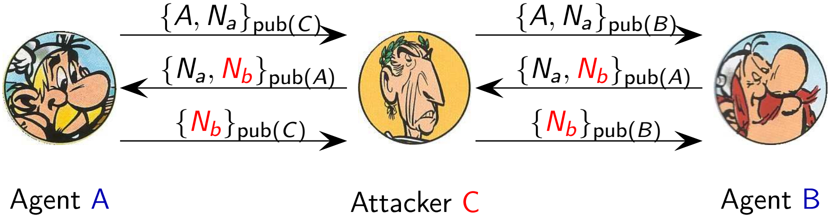
$$
\begin{array}{lcl}
\mathsf{A} \to \mathsf{B} & : & \{A, N_a\}_{\mathsf{pub}(B)} \\
\mathsf{B} \to \mathsf{A} & : & \{N_a, N_b\}_{\mathsf{pub}(A)} \\
\mathsf{A} \to \mathsf{B} & : & \{N_b\}_{\mathsf{pub}(B)}
\end{array}
$$

Man in the middle attack



$$
\begin{array}{lll}
\mathrm{A} \rightarrow \mathrm{B} & : & \{A, N_a\}_{\mathrm{pub}(B)} \\
\mathrm{B} \rightarrow \mathrm{A} & : & \{N_a, N_b\}_{\mathrm{pub}(A)} \\
\mathrm{A} \rightarrow \mathrm{B} & : & \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$

Man in the middle attack



Attack

- the intruder knows N_b,
- When B finishes his session (apparently with A), A has never talked with B.

$$
\begin{array}{lcl}
A \rightarrow B & : & \{A, N_a\}_{\mathrm{pub}(B)} \\
B \rightarrow A & : & \{N_a, N_b\}_{\mathrm{pub}(A)} \\
A \rightarrow B & : & \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$

Needham Schroeder Lowe protocol

A fixed version of the Needham Schroeder public key protocol.

$$\begin{array}{lll} \mathsf{A} \to \mathsf{B} & : & \{A, N_a\}_{\mathsf{pub}(B)} \\ \mathsf{B} \to \mathsf{A} & : & \{N_a, N_b, \color{red}{B}\}_{\mathsf{pub}(A)} \\ \mathsf{A} \to \mathsf{B} & : & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

$\longrightarrow$ the responder's identity has been added to the second message

Outline for today

Security problem for a bounded number of sessions

$\longrightarrow$ *i.e.* processes with no replication

... using the constraint solving approach

Two main kind of security properties:

1. trace-based security properties (*e.g.* secrecy, authentication, ...)
2. equivalence-based security properties (*e.g.* anonymity, untraceability, ...)

Running examples:

1. Needham-Schroeder protocol
2. BAC protocol used in the e-passport application

Part I

Trace-based security properties

Reminder

Syntax : $P, Q \quad := \quad$

0	null process
$\mathsf{in}(c, x).P$	input
$\mathsf{out}(c, u).P$	output
if $u = v$ then P else Q	conditional
$P \mid Q$	parallel composition
$!P$	replication
new $n.P$	fresh name generation

Confidentiality for process P w.r.t. secret s

For all processes A such that $A \mid P \rightarrow^* Q$, we have that Q is not of the form $C[\mathsf{out}(c, s).Q']$ with c public.

$\longrightarrow$ In other word, s should not be deducible by the attacker

Confidentiality using the constraint solving approach

$\longrightarrow$ for a bounded number of sessions

Two main steps:

1. A symbolic exploration of all the possible traces

   The infinite number of possible traces (*i.e.* experiment) are represented by a finite set of constraint systems

   $\longrightarrow$ this set can be huge (exponential on the number of sessions) ... but some optimizations are used to reduce this number

2. A decision procedure for deciding whether a constraint system has a solution or not.

   $\longrightarrow$ this algorithm works quite well

Confidentiality via constraint solving

Constraint systems are used to specify confidentiality under a particular scenario.

Protocol rules

- a particular interleaving -

$\mathsf{in}(u_1);$

$\mathsf{out}(v_1); \mathsf{in}(u_2);$

$\ldots$

$\mathsf{out}(v_n)$

Constraint System

$$\mathcal{C} = \begin{cases} T_0 \overset{?}{\vdash} u_1 \\ T_0, v_1 \overset{?}{\vdash} u_2 \\ \quad \ldots \\ T_0, v_1, .., v_n \overset{?}{\vdash} s \end{cases}$$

Confidentiality via constraint solving

Constraint systems are used to specify confidentiality under a particular scenario.

Protocol rules

- a particular interleaving -

$\mathsf{in}(u_1);$

$\mathsf{out}(v_1); \mathsf{in}(u_2);$

$\ldots$

$\mathsf{out}(v_n)$

Constraint System

$$\mathcal{C} = \begin{cases} T_0 \overset{?}{\vdash} u_1 \\ T_0, v_1 \overset{?}{\vdash} u_2 \\ \quad \ldots \\ T_0, v_1, .., v_n \overset{?}{\vdash} s \end{cases}$$

Solution of a constraint system $\mathcal{C}$

A substitution σ such that

for every $T \overset{?}{\vdash} u \in \mathcal{C}$, $u\sigma$ *is deducible from* $T\sigma$.

for every $u = v \in \mathcal{C}$ *(resp.* $u \neq v$*)*, $u\sigma =_{\mathsf{E}} v\sigma$ *(resp.* $u\sigma \neq_{\mathsf{E}} v\sigma$*)*

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\textit{new } n_a. \quad \mathsf{out}(\{a, n_a\}_{\mathsf{pub}(c)}). \quad \mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)}). \quad \mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(c)})$$

Role B played by b (apparently) with a:

$$\mathsf{in}(\{a, y_{n_a}\}_{\mathsf{pub}(b)}). \quad \textit{new } n_b. \quad \mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(a)})$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a. \quad \underset{1}{\mathsf{out}(\{a, n_a\}_{\mathsf{pub}(c)})}. \quad \underset{4}{\mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)})}. \quad \underset{5}{\mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(c)})}$$

Role B played by b (apparently) with a:

$$\underset{2}{\mathsf{in}(\{a, y_{n_a}\}_{\mathsf{pub}(b)})}. \quad \mathit{new}\ n_b. \quad \underset{3}{\mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(a)})}$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\underset{}{\textit{new } n_a.} \quad \underset{1}{\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(c)}).} \quad \underset{4}{\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)}).} \quad \underset{5}{\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(c)})}$$

Role B played by b (apparently) with a:

$$\underset{2}{\mathrm{in}(\{a, y_{n_a}\}_{\mathrm{pub}(b)}).} \quad \textit{new } n_b. \quad \underset{3}{\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(a)})}$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathrm{priv}(c)\}$:

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a.\quad \underset{1}{\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(c)})}.\quad \underset{4}{\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})}.\quad \underset{5}{\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(c)})}$$

Role B played by b (apparently) with a:

$$\underset{2}{\mathrm{in}(\{a, y_{n_a}\}_{\mathrm{pub}(b)})}.\quad \mathit{new}\ n_b.\quad \underset{3}{\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(a)})}$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathrm{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)}$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a. \quad \underset{1}{\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(c)})}. \quad \underset{4}{\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})}. \quad \underset{5}{\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(c)})}$$

Role B played by b (apparently) with a:

$$\underset{2}{\mathrm{in}(\{a, y_{n_a}\}_{\mathrm{pub}(b)})}. \quad \mathit{new}\ n_b. \quad \underset{3}{\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(a)})}$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathrm{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)} \overset{?}{\vdash} \{a, y_{n_a}\}_{\mathrm{pub}(b)}$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a.\quad \underset{1}{\mathsf{out}(\{a, n_a\}_{\mathsf{pub}(c)})}.\quad \underset{4}{\mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)})}.\quad \underset{5}{\mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(c)})}$$

Role B played by b (apparently) with a:

$$\underset{2}{\mathsf{in}(\{a, y_{n_a}\}_{\mathsf{pub}(b)})}.\quad \mathit{new}\ n_b.\quad \underset{3}{\mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(a)})}$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathsf{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\mathsf{pub}(c)} \overset{?}{\vdash} \{a, y_{n_a}\}_{\mathsf{pub}(b)}$$

$$T_0,\ \{a, n_a\}_{\mathsf{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(a)}$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a. \quad \underset{1}{\text{out}(\{a, n_a\}_{\text{pub}(c)})}. \quad \underset{4}{\text{in}(\{n_a, x_{n_b}\}_{\text{pub}(a)})}. \quad \underset{5}{\text{out}(\{x_{n_b}\}_{\text{pub}(c)})}$$

Role B played by b (apparently) with a:

$$\underset{2}{\text{in}(\{a, y_{n_a}\}_{\text{pub}(b)})}. \quad \mathit{new}\ n_b. \quad \underset{3}{\text{out}(\{y_{n_a}, n_b\}_{\text{pub}(a)})}$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \text{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\text{pub}(c)} \overset{?}{\vdash} \{a, y_{n_a}\}_{\text{pub}(b)}$$

$$T_0,\ \{a, n_a\}_{\text{pub}(c)},\ \{y_{n_a}, n_b\}_{\text{pub}(a)} \overset{?}{\vdash} \{n_a, x_{n_b}\}_{\text{pub}(a)}$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a. \quad \underset{1}{\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(c)})}. \quad \underset{4}{\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})}. \quad \underset{5}{\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(c)})}$$

Role B played by b (apparently) with a:

$$\underset{2}{\mathrm{in}(\{a, y_{n_a}\}_{\mathrm{pub}(b)})}. \quad \mathit{new}\ n_b. \quad \underset{3}{\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(a)})}$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathrm{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)} \overset{?}{\vdash} \{a, y_{n_a}\}_{\mathrm{pub}(b)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)} \overset{?}{\vdash} \{n_a, x_{n_b}\}_{\mathrm{pub}(a)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)},\ \{x_{n_b}\}_{\mathrm{pub}(c)}$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a.\quad \mathrm{out}(\{a, n_a\}_{\mathrm{pub}(c)}).\quad \mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)}).\quad \mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(c)})$$

Role B played by b (apparently) with a:

$$\mathrm{in}(\{a, y_{n_a}\}_{\mathrm{pub}(b)}).\quad \mathit{new}\ n_b.\quad \mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(a)})$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathrm{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)} \overset{?}{\vdash} \{a, y_{n_a}\}_{\mathrm{pub}(b)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)} \overset{?}{\vdash} \{n_a, x_{n_b}\}_{\mathrm{pub}(a)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)},\ \{x_{n_b}\}_{\mathrm{pub}(c)} \overset{?}{\vdash} n_b$$

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a. \quad \mathrm{out}(\{a, n_a\}_{\mathrm{pub}(c)}). \quad \mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)}). \quad \mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(c)})$$

Role B played by b (apparently) with a:

$$\mathrm{in}(\{a, y_{n_a}\}_{\mathrm{pub}(b)}). \quad \mathit{new}\ n_b. \quad \mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(a)})$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathrm{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)} \stackrel{?}{\vdash} \{a, y_{n_a}\}_{\mathrm{pub}(b)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)} \stackrel{?}{\vdash} \{n_a, x_{n_b}\}_{\mathrm{pub}(a)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)},\ \{x_{n_b}\}_{\mathrm{pub}(c)} \stackrel{?}{\vdash} n_b$$

Does this constraint system have a solution?

Going back to the Needham-Schroeder's protocol

Role A played by a with the attacker c:

$$\mathit{new}\ n_a. \quad \mathrm{out}(\{a, n_a\}_{\mathrm{pub}(c)}). \quad \mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)}). \quad \mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(c)})$$

Role B played by b (apparently) with a:

$$\mathrm{in}(\{a, y_{n_a}\}_{\mathrm{pub}(b)}). \quad \mathit{new}\ n_b. \quad \mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(a)})$$

Constraint system: (secrecy of n_b) with $T_0 = \{a, b, c, \mathrm{priv}(c)\}$:

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)} \overset{?}{\vdash} \{a, y_{n_a}\}_{\mathrm{pub}(b)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)} \overset{?}{\vdash} \{n_a, x_{n_b}\}_{\mathrm{pub}(a)}$$

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(c)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(a)},\ \{x_{n_b}\}_{\mathrm{pub}(c)} \overset{?}{\vdash} n_b$$

Does this constraint system have a solution?

$\longrightarrow$ Yes ! $\quad \sigma = \{y_a \mapsto a,\ \ y_{n_a} \mapsto n_a,\ \ x_{n_b} \mapsto n_b\}$

Going back to the Denning Sacco protocol

$$\begin{array}{lcl} A \to B & : & \mathsf{aenc}(\mathsf{sign}(k, \mathsf{priv}(A)), \mathsf{pub}(B)) \\ B \to A & : & \mathsf{senc}(s, k) \end{array}$$

One possible interleaving:

$$\mathsf{out}(\mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)))$$
$$\mathsf{in}(\mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb))); \mathsf{out}(\mathsf{senc}(s, x))$$

$$\begin{array}{lcl} A \to B & : & \mathsf{aenc}(\mathsf{sign}(k, \mathsf{priv}(A)), \mathsf{pub}(B)) \\ B \to A & : & \mathsf{senc}(s, k) \end{array}$$

One possible interleaving:

$$\mathsf{out}(\mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)))$$
$$\mathsf{in}(\mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb))); \mathsf{out}(\mathsf{senc}(s, x))$$

The associated constraint system is:

$$\begin{array}{rcl} T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) & \overset{?}{\vdash} & \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb)) \\ T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc));\ \mathsf{senc}(s, x) & \overset{?}{\vdash} & s \end{array}$$

with $T_0 = \{\mathsf{pk}(ska),\ \mathsf{pk}(skb);\ skc\}$.

Going back to the Denning Sacco protocol

$$\begin{array}{rcl} A \to B & : & \mathsf{aenc}(\mathsf{sign}(k, \mathsf{priv}(A)), \mathsf{pub}(B)) \\ B \to A & : & \mathsf{senc}(s, k) \end{array}$$

One possible interleaving:

$$\mathsf{out}(\mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)))$$
$$\mathsf{in}(\mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb))); \mathsf{out}(\mathsf{senc}(s, x))$$

The associated constraint system is:

$$\begin{array}{rcl} T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) & \overset{?}{\vdash} & \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb)) \\ T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc));\ \mathsf{senc}(s, x) & \overset{?}{\vdash} & s \end{array}$$

with $T_0 = \{\mathsf{pk}(ska),\ \mathsf{pk}(skb);\ skc\}$.

Does this constraint system have a solution?

Going back to the Denning Sacco protocol

$$\begin{array}{lcl} A \to B & : & \mathsf{aenc}(\mathsf{sign}(k, \mathsf{priv}(A)), \mathsf{pub}(B)) \\ B \to A & : & \mathsf{senc}(s, k) \end{array}$$

One possible interleaving:

$$\mathsf{out}(\mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)))$$
$$\mathsf{in}(\mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb))); \mathsf{out}(\mathsf{senc}(s, x))$$

The associated constraint system is:

$$\begin{array}{rcl} T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) & \overset{?}{\vdash} & \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb)) \\ T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc));\ \mathsf{senc}(s, x) & \overset{?}{\vdash} & s \end{array}$$

with $T_0 = \{\mathsf{pk}(ska),\ \mathsf{pk}(skb);\ skc\}$.

Does this constraint system have a solution?

Yes ! $x \to k$

The general case: is the constraint system $\mathcal{C}$ satisfiable?

Main idea: simplify them until reaching $\perp$ or solved forms

Constraint system in solved form

$$\mathcal{C} = \begin{cases} T_0 \overset{?}{\vdash} x_0 \\ T_0 \cup T_1 \overset{?}{\vdash} x_1 \\ \quad \dots \\ T_0 \cup T_1 \ldots \cup T_n \overset{?}{\vdash} x_n \end{cases}$$

Question

Is there a solution to such a system ?

The general case: is the constraint system $\mathcal{C}$ satisfiable?

Main idea: simplify them until reaching $\perp$ or solved forms

Constraint system in solved form

$$\mathcal{C} = \begin{cases} T_0 \overset{?}{\vdash} x_0 \\ T_0 \cup T_1 \overset{?}{\vdash} x_1 \\ \quad \dots \\ T_0 \cup T_1 \ldots \cup T_n \overset{?}{\vdash} x_n \end{cases}$$

Question

Is there a solution to such a system ?

Of course, yes ! Choose $u_0 \in T_0$, and consider the substitution:

$$\sigma = \{x_0 \mapsto u_0, \ldots, x_n \mapsto u_0\}$$

Simplification rules

$\longrightarrow$ these rules deal with pairs and symmetric encryption only

$$\mathsf{R}_{\mathsf{ax}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \mathcal{C} \quad \text{if } T \cup \{x \mid T' \overset{?}{\vdash} x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

$$\mathsf{R}_{\mathsf{unif}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow_{\sigma} \quad \mathcal{C}\sigma \wedge T\sigma \overset{?}{\vdash} u\sigma$$
$$\text{if } \sigma = \mathrm{mgu}(t_1, t_2) \text{ where } t_1, t_2 \in st(T) \cup \{u\}$$

$$\mathsf{R}_{\mathsf{fail}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \bot \quad \text{if } vars(T \cup \{u\}) = \emptyset \text{ and } T \not\vdash u$$

$$\mathsf{R}_{\mathsf{f}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} f(u_1, u_2) \rightsquigarrow \mathcal{C} \wedge T \overset{?}{\vdash} u_1 \wedge T \overset{?}{\vdash} u_2 \quad f \in \{\langle\rangle, \mathrm{senc}\}$$

Applying rule $\mathsf{R_f}$

$$\mathsf{R_f}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} \mathsf{f}(u_1, u_2) \rightsquigarrow \mathcal{C} \wedge T \overset{?}{\vdash} u_1 \wedge T \overset{?}{\vdash} u_2$$

Example:

$$T_0; \; \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) \overset{?}{\vdash} \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb))$$

Applying rule $\mathsf{R_f}$

$$\mathsf{R_f}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} \mathsf{f}(u_1, u_2) \rightsquigarrow \mathcal{C} \wedge T \overset{?}{\vdash} u_1 \wedge T \overset{?}{\vdash} u_2$$

Example:

$$T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) \overset{?}{\vdash} \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb))$$

$$\rightsquigarrow \begin{cases} T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) \overset{?}{\vdash} \mathsf{sign}(x, ska) \\ T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) \overset{?}{\vdash} \mathsf{pk}(skb) \end{cases}$$

Applying rule $\mathsf{R}_{\mathsf{unif}}$

$$\mathsf{R}_{\mathsf{unif}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow_{\sigma} \quad \mathcal{C}\sigma \wedge T\sigma \overset{?}{\vdash} u\sigma$$
$$\text{if } \sigma = \mathrm{mgu}(t_1, t_2) \text{ where } t_1, t_2 \in st(T) \cup \{u\}$$

Example:

$$\begin{cases} T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} & \mathrm{sign}(x, ska) \\ T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} & \mathrm{pk}(skb) \end{cases}$$

Applying rule $\mathsf{R_{unif}}$

$$\mathsf{R_{unif}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow_{\sigma} \quad \mathcal{C}\sigma \wedge T\sigma \overset{?}{\vdash} u\sigma$$

$$\text{if } \sigma = \mathrm{mgu}(t_1, t_2) \text{ where } t_1, t_2 \in st(T) \cup \{u\}$$

Example:

$$\begin{cases} T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} \quad \mathrm{sign}(x, ska) \\ T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} \quad \mathrm{pk}(skb) \end{cases}$$

$$\rightsquigarrow \begin{cases} T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} \quad \mathrm{sign}(k, ska) \\ T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} \quad \mathrm{pk}(skb) \end{cases}$$

Applying rule R_{ax}

$$\mathsf{R}_{\mathsf{ax}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \mathcal{C} \qquad \text{if } T \cup \{x \mid T' \overset{?}{\vdash} x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

Example: (assuming that skc and $\mathrm{pk}(skb)$ are in T_0)

$$\begin{cases} T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} & \mathrm{sign}(k, ska) \\ T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} & \mathrm{pk}(skb) \end{cases}$$

Applying rule R_{ax}

$$\mathsf{R}_{\mathsf{ax}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \mathcal{C} \qquad \text{if } T \cup \{x \mid T' \overset{?}{\vdash} x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

Example: (assuming that *skc* and $\mathsf{pk}(skb)$ are in T_0)

$$\begin{cases} T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) & \overset{?}{\vdash} & \mathsf{sign}(k, ska) \\ T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) & \overset{?}{\vdash} & \mathsf{pk}(skb) \end{cases}$$

$$\rightsquigarrow \left\{ T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) \overset{?}{\vdash} \mathsf{sign}(k, ska) \right.$$

Applying rule R_{ax}

$$R_{ax}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \mathcal{C} \qquad \text{if } T \cup \{x \mid T' \overset{?}{\vdash} x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

Example: (assuming that *skc* and $\mathrm{pk}(skb)$ are in T_0)

$$\begin{cases} T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} & \mathrm{sign}(k, ska) \\ T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) & \overset{?}{\vdash} & \mathrm{pk}(skb) \end{cases}$$

$$\rightsquigarrow \left\{ T_0;\ \mathrm{aenc}(\mathrm{sign}(k, ska), \mathrm{pk}(skc)) \overset{?}{\vdash} \mathrm{sign}(k, ska) \right.$$

$\rightsquigarrow \quad \emptyset \quad$ (empty constraint system)

Exercice - still about the Denning Sacco protocol

Exercise

Reach a solved form starting with the constraint system:

$$
\begin{array}{rcl}
T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc)) & \stackrel{?}{\vdash} & \mathsf{aenc}(\mathsf{sign}(x, ska), \mathsf{pk}(skb)) \\
T_0;\ \mathsf{aenc}(\mathsf{sign}(k, ska), \mathsf{pk}(skc));\ \mathsf{senc}(s, x) & \stackrel{?}{\vdash} & s
\end{array}
$$

Results on the simplification rules

$$\mathsf{R}_{\mathsf{ax}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \mathcal{C} \quad \text{if } T \cup \{x \mid T' \overset{?}{\vdash} x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

$$\mathsf{R}_{\mathsf{unif}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow_\sigma \quad \mathcal{C}\sigma \wedge T\sigma \overset{?}{\vdash} u\sigma$$
$$\text{if } \sigma = \mathrm{mgu}(t_1, t_2) \text{ where } t_1, t_2 \in st(T) \cup \{u\}$$

$$\mathsf{R}_{\mathsf{fail}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \bot \quad \text{if } vars(T \cup \{u\}) = \emptyset \text{ and } T \not\vdash u$$

$$\mathsf{R}_{\mathsf{f}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} f(u_1, u_2) \rightsquigarrow \mathcal{C} \wedge T \overset{?}{\vdash} u_1 \wedge T \overset{?}{\vdash} u_2 \quad f \in \{\langle\rangle, \mathrm{senc}\}$$

Given a (well-formed) constraint system $\mathcal{C}$:

Soundness

If $\mathcal{C} \rightsquigarrow^*_\sigma \mathcal{C}'$ and θ solution of $\mathcal{C}'$ then $\sigma\theta$ is a solution of $\mathcal{C}$.

$\longrightarrow$ easy to show

Results on the simplification rules

$$\mathsf{R}_{\mathsf{ax}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \mathcal{C} \quad \text{if } T \cup \{x \mid T' \overset{?}{\vdash} x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

$$\mathsf{R}_{\mathsf{unif}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow_\sigma \quad \mathcal{C}\sigma \wedge T\sigma \overset{?}{\vdash} u\sigma$$
$$\text{if } \sigma = \mathrm{mgu}(t_1, t_2) \text{ where } t_1, t_2 \in st(T) \cup \{u\}$$

$$\mathsf{R}_{\mathsf{fail}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \perp \quad \text{if } vars(T \cup \{u\}) = \emptyset \text{ and } T \not\vdash u$$

$$\mathsf{R}_{\mathsf{f}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} f(u_1, u_2) \rightsquigarrow \mathcal{C} \wedge T \overset{?}{\vdash} u_1 \wedge T \overset{?}{\vdash} u_2 \quad f \in \{\langle\rangle, \mathrm{senc}\}$$

Given a (well-formed) constraint system $\mathcal{C}$:

Termination

There is no infinite chain $\mathcal{C} \rightsquigarrow_{\sigma_1} \mathcal{C}_1 \ldots \rightsquigarrow_{\sigma_n} \mathcal{C}_n$.

$\longrightarrow$ using the lexicographic order (number of var, size of rhs)

Results on the simplification rules

$$\mathsf{R}_{\mathsf{ax}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \mathcal{C} \quad \text{if } T \cup \{x \mid T' \overset{?}{\vdash} x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

$$\mathsf{R}_{\mathsf{unif}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow_\sigma \quad \mathcal{C}\sigma \wedge T\sigma \overset{?}{\vdash} u\sigma$$
$$\text{if } \sigma = \mathrm{mgu}(t_1, t_2) \text{ where } t_1, t_2 \in st(T) \cup \{u\}$$

$$\mathsf{R}_{\mathsf{fail}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} u \quad \rightsquigarrow \quad \bot \qquad \text{if } vars(T \cup \{u\}) = \emptyset \text{ and } T \not\vdash u$$

$$\mathsf{R}_{\mathsf{f}}: \quad \mathcal{C} \wedge T \overset{?}{\vdash} f(u_1, u_2) \rightsquigarrow \mathcal{C} \wedge T \overset{?}{\vdash} u_1 \wedge T \overset{?}{\vdash} u_2 \quad f \in \{\langle\rangle, \mathrm{senc}\}$$

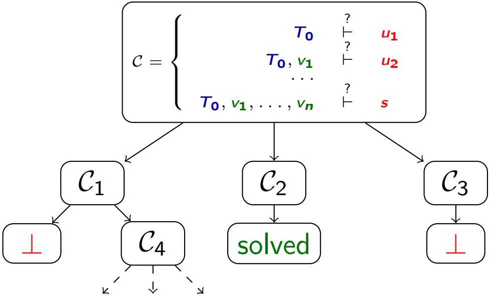
Given a (well-formed) constraint system $\mathcal{C}$:

Completeness

If θ is a solution of $\mathcal{C}$ then there exists $\mathcal{C}'$ and θ' such that $\mathcal{C} \rightsquigarrow^*_\sigma \mathcal{C}'$, θ' is a solution of $\mathcal{C}'$, and $\theta = \sigma\theta'$.

$\longrightarrow$ more involved to show

Main idea of the procedure:



$\longrightarrow$ this gives us a symbolic representation of all the solutions.

Main result

Theorem

Deciding confidentiality for a bounded number of sessions is decidable for classical primitives (actually in co-NP).

Exercise: NP-hardness can be shown by encoding 3-SAT

Main result

Theorem

Deciding confidentiality for a bounded number of sessions is decidable for classical primitives (actually in co-NP).

Exercise: NP-hardness can be shown by encoding 3-SAT

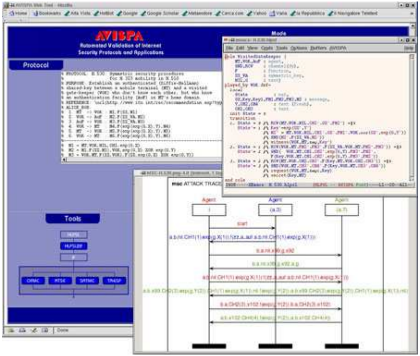
Some extensions that already exist:

1. disequality tests (protocol with else branches)
2. more primitives: asymmetric encryption, blind signature, exclusive-or, ...

Avantssar platform

This approach has been implemented in the Avantssar Platform.

http://www.avantssar.eu



⟶ Typically concludes within few seconds over the flawed protocols of the Clark/Jacob library .

Part II

Equivalence-based security properties

Electronic passport

⟶ studied in [Arapinis *et al.*, 10]

An electronic passport is a passport with an RFID tag embedded in it.

The RFID tag stores:

- the information printed on your passport,
- a JPEG copy of your picture.

Electronic passport

⟶ studied in [Arapinis *et al.*, 10]

An electronic passport is a passport with an RFID tag embedded in it.

The RFID tag stores:

- the information printed on your passport,
- a JPEG copy of your picture.

The Basic Access Control (BAC) protocol is a key establishment protocol that has been designed to also ensure unlinkability.

ISO/IEC standard 15408

Unlinkability aims to ensure *that a user may make multiple uses of a service or resource without others being able to link these uses together.*

BAC protocol

Passport
(K_E, K_M)

Reader
(K_E, K_M)

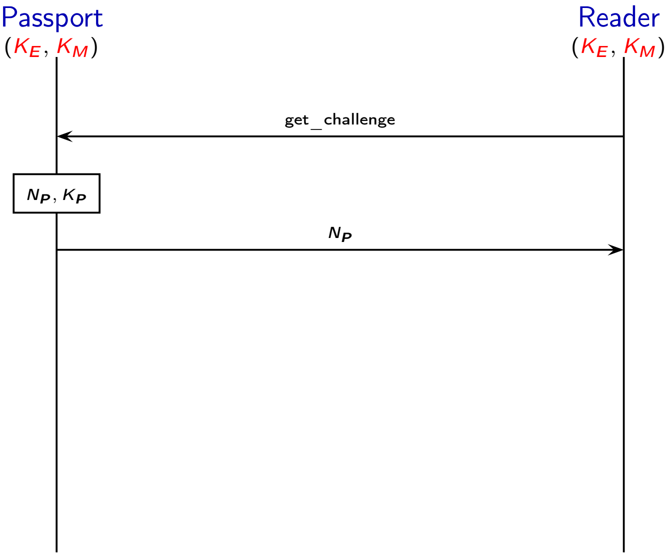
BAC protocol

Passport
(K_E, K_M)

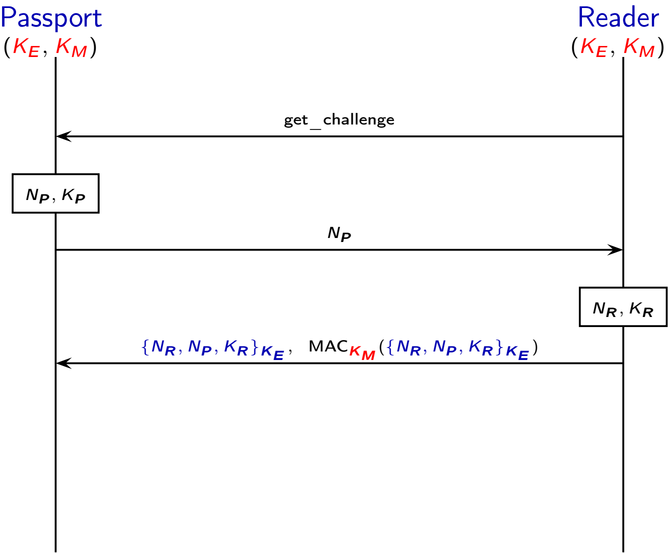
Reader
(K_E, K_M)

$\text{Passport} \xleftarrow{\text{get\_challenge}} \text{Reader}$

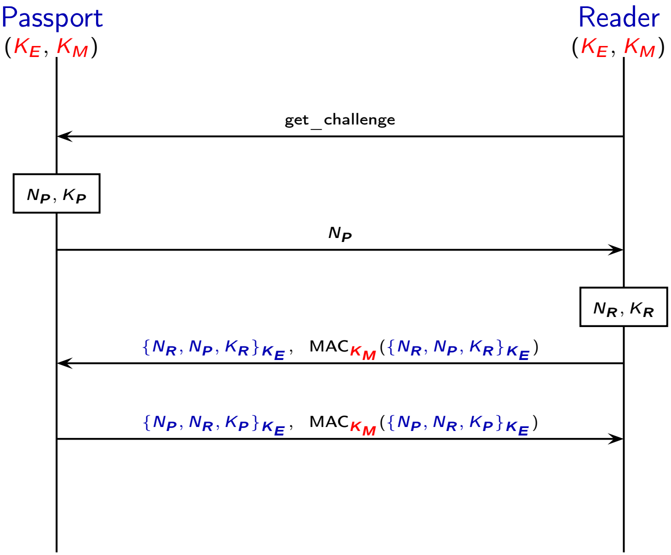
BAC protocol



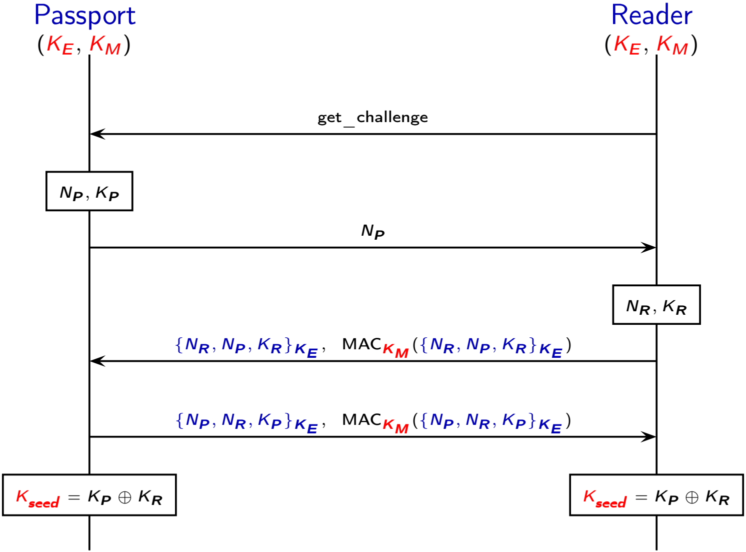
BAC protocol



BAC protocol



BAC protocol



BAC protocol as a process

Cryptographic primitives are modelled using function symbols

- encryption/decryption: senc/2, sdec/2
- concatenation/projections: $\langle\, ,\, \rangle/2$, $\mathsf{proj}_1/1$, $\mathsf{proj}_2/1$
- mac construction: mac/2

$\longrightarrow \quad \mathsf{sdec}(\mathsf{senc}(x,y),y) = x, \quad \mathsf{proj}_1(\langle x,y\rangle) = x, \quad \mathsf{proj}_2(\langle x,y\rangle) = y.$

Nonces n_r, n_p, and keys k_r, k_p, k_e, k_m are modelled using names

BAC protocol as a process

Cryptographic primitives are modelled using function symbols

- encryption/decryption: senc/2, sdec/2
- concatenation/projections: $\langle\ ,\ \rangle/2$, $\mathsf{proj}_1/1$, $\mathsf{proj}_2/1$
- mac construction: mac/2

$\longrightarrow \quad \mathsf{sdec}(\mathsf{senc}(x,y),y) = x, \quad \mathsf{proj}_1(\langle x,y\rangle) = x, \quad \mathsf{proj}_2(\langle x,y\rangle) = y.$

Nonces n_r, n_p, and keys k_r, k_p, k_e, k_m are modelled using names

Modelling Passport's role

$$
\begin{array}{l}
P_{\mathrm{BAC}}(k_E, k_M) \ = \textit{new } n_P.\textit{new } k_P.\mathsf{out}(n_P).\mathsf{in}(\langle z_E, z_M\rangle).\\
\quad \texttt{if } z_M = \mathsf{mac}(z_E, k_M) \texttt{ then if } n_P = \mathsf{proj}_1(\mathsf{proj}_2(\mathsf{sdec}(z_E, k_E)))\\
\qquad\qquad\qquad\qquad\qquad \texttt{then}\ \mathsf{out}(\langle m, \mathsf{mac}(m, k_M)\rangle)\\
\qquad\qquad\qquad\qquad\qquad \texttt{else}\ 0\\
\qquad\qquad\qquad\quad \texttt{else}\ 0
\end{array}
$$

where $m = \mathsf{senc}(\langle n_P, \langle \mathsf{proj}_1(z_E), k_P\rangle\rangle, k_E)$.

What does unlinkability mean?

Informally, an observer/attacker can not observe the difference between the two following situations:

1. a situation where the same passport may be used twice (or even more);
2. a situation where each passport is used at most once.

What does unlinkability mean?

Informally, an observer/attacker can not observe the difference between the two following situations:

1. a situation where the same passport may be used twice (or even more);
2. a situation where each passport is used at most once.

More formally,

$$!new\ ke.new\ km.(!P_{\mathrm{BAC}} \mid !R_{\mathrm{BAC}}) \overset{?}{\approx} !new\ ke.new\ km.(P_{\mathrm{BAC}} \mid !R_{\mathrm{BAC}})$$

many sessions for each passport

only one session for each passport

(we still have to formalize the notion of equivalence)

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

testing equivalence between P and Q, $P \approx Q$

for all processes A, we have that:

$$(A \mid P) \Downarrow_c \text{ if, and only if, } (A \mid Q) \Downarrow_c$$

where $R \Downarrow_c$ means that R can evolve and emits on public channel c.

Privacy-type properties are modelled as equivalence-based properties

testing equivalence between P and Q, $P \approx Q$

for all processes A, we have that:

$$(A \mid P) \Downarrow_c \text{ if, and only if, } (A \mid Q) \Downarrow_c$$

where $R \Downarrow_c$ means that R can evolve and emits on public channel c.

Example 1: $\mathrm{out}(a, s) \stackrel{?}{\approx} \mathrm{out}(a, s')$

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

testing equivalence between P and Q, $P \approx Q$

for all processes A, we have that:

$$(A \mid P) \Downarrow_c \text{ if, and only if, } (A \mid Q) \Downarrow_c$$

where $R \Downarrow_c$ means that R can evolve and emits on public channel c.

Example 1: $\quad \mathrm{out}(a, s) \not\approx \mathrm{out}(a, s')$

$$\longrightarrow \quad A = \mathrm{in}(a, x).\mathrm{if}\ x = s\ \mathrm{then}\ \mathrm{out}(c, ok)$$

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

testing equivalence between P and Q, $P \approx Q$

for all processes A, we have that:

$$(A \mid P) \Downarrow_c \text{ if, and only if, } (A \mid Q) \Downarrow_c$$

where $R \Downarrow_c$ means that R can evolve and emits on public channel c.

Example 2:

$$\mathit{new}\ s.\mathrm{out}(a, \mathrm{senc}(s, k)).\mathrm{out}(a, \mathrm{senc}(s, k'))$$
$$\stackrel{?}{\approx}$$
$$\mathit{new}\ s, s'.\mathrm{out}(a, \mathrm{senc}(s, k)).\mathrm{out}(a, \mathrm{senc}(s', k'))$$

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

testing equivalence between P and Q, $P \approx Q$

for all processes A, we have that:

$$(A \mid P) \Downarrow_c \text{ if, and only if, } (A \mid Q) \Downarrow_c$$

where $R \Downarrow_c$ means that R can evolve and emits on public channel c.

Example 2:

$$\begin{array}{c} \mathit{new}\ s.\mathsf{out}(a, \mathsf{senc}(s, k)).\mathsf{out}(a, \mathsf{senc}(s, k')) \\ \not\approx \\ \mathit{new}\ s, s'.\mathsf{out}(a, \mathsf{senc}(s, k)).\mathsf{out}(a, \mathsf{senc}(s', k')) \end{array}$$

$$\longrightarrow A = \mathsf{in}(a, x).\mathsf{in}(a, y).\mathsf{if}\ (\mathsf{sdec}(x, k) = \mathsf{sdec}(y, k'))\ \mathsf{then}\ \mathsf{out}(c, ok)$$

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

testing equivalence between P and Q, $P \approx Q$

for all processes A, we have that:

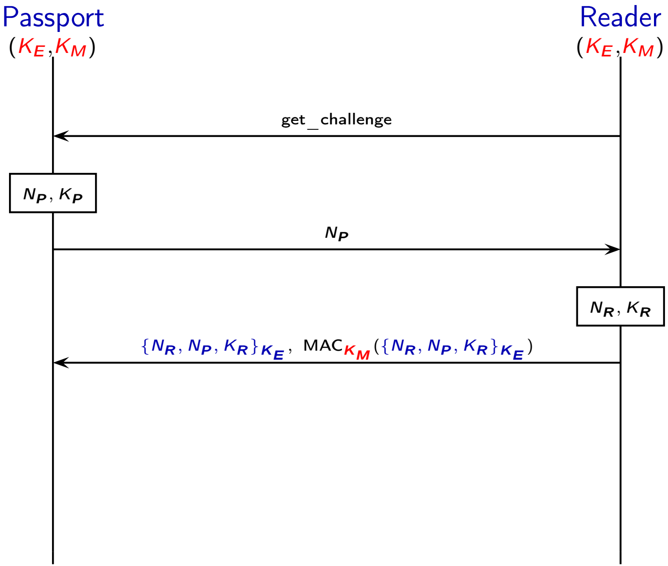
$$(A \mid P) \Downarrow_c \text{ if, and only if, } (A \mid Q) \Downarrow_c$$

where $R \Downarrow_c$ means that R can evolve and emits on public channel c.

Exercise: Are the two following processes in testing equivalence?

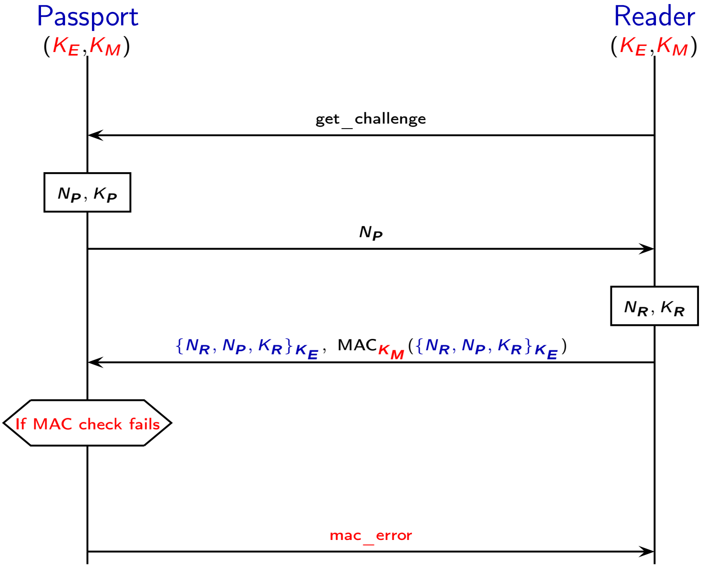
$$\mathit{new}\, s.\mathsf{out}(a, s) \stackrel{?}{\approx} \mathit{new}\, s.\mathit{new}\, k.\mathsf{out}(a, \mathsf{enc}(s, k))$$

$\longrightarrow$ the passport must reply to all received messages.



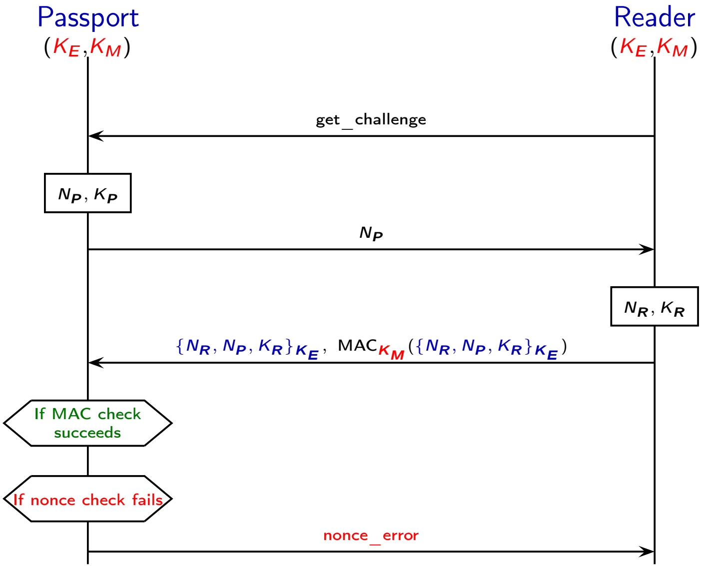
French electronic passport

⟶ the passport must reply to all received messages.



French electronic passport

$\longrightarrow$ the passport must reply to all received messages.



BAC protocol (French version) as a process

Cryptographic primitives are modelled as usual using function symbols

$\longrightarrow \quad \mathsf{sdec}(\mathsf{senc}(x,y),y) = x, \quad \mathsf{proj}_1(\langle x,y\rangle) = x, \quad \mathsf{proj}_2(\langle x,y\rangle) = y.$

Nonces n_r, n_p, and keys k_r, k_p, k_e, k_m are modelled using names

Error messages are modelled using constants *mac_error* and *nonce_error*.

BAC protocol (French version) as a process

Cryptographic primitives are modelled as usual using function symbols

$\longrightarrow \quad \mathsf{sdec}(\mathsf{senc}(x, y), y) = x, \quad \mathsf{proj}_1(\langle x, y\rangle) = x, \quad \mathsf{proj}_2(\langle x, y\rangle) = y.$

Nonces n_r, n_p, and keys k_r, k_p, k_e, k_m are modelled using names

Error messages are modelled using constants *mac_error* and *nonce_error*.

Modelling Passport's role

$$
\begin{array}{l}
P_{\mathsf{BAC}}(k_E, k_M) \; = \mathit{new}\; n_P.\mathit{new}\; k_P.\mathsf{out}(n_P).\mathsf{in}(\langle z_E, z_M\rangle). \\
\quad \mathtt{if}\; z_M = \mathsf{mac}(z_E, k_M)\; \mathtt{then}\; \mathtt{if}\; n_P = \mathsf{proj}_1(\mathsf{proj}_2(\mathsf{sdec}(z_E, k_E))) \\
\qquad\qquad\qquad\qquad\qquad\qquad \mathtt{then}\; \mathsf{out}(\langle m, \mathsf{mac}(m, k_M)\rangle) \\
\qquad\qquad\qquad\qquad\qquad\qquad \mathtt{else}\; \mathsf{out}(\mathit{nonce\_error}) \\
\qquad\qquad\qquad\qquad\qquad \mathtt{else}\; \mathsf{out}(\mathit{mac\_error})
\end{array}
$$

where $m = \mathsf{senc}(\langle n_P, \langle \mathsf{proj}_1(z_E), k_P\rangle\rangle, k_E)$.

An attack on the French passport

Attack against unlinkability [Chothia & Smirnov, 10]

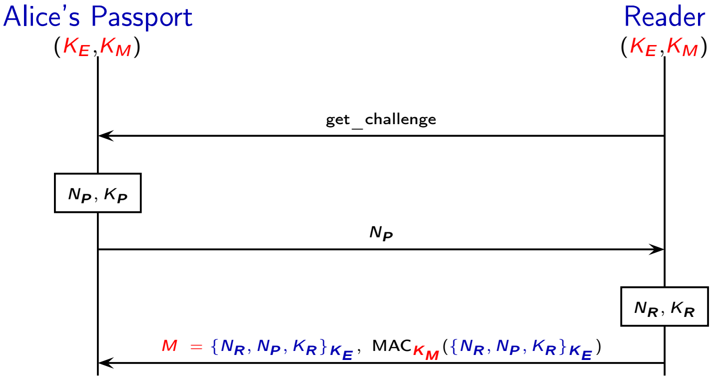
An attacker can track a French passport, provided he has once witnessed a successful authentication.

An attack on the French passport

Attack against unlinkability [Chothia & Smirnov, 10]

An attacker can track a French passport, provided he has once witnessed a successful authentication.

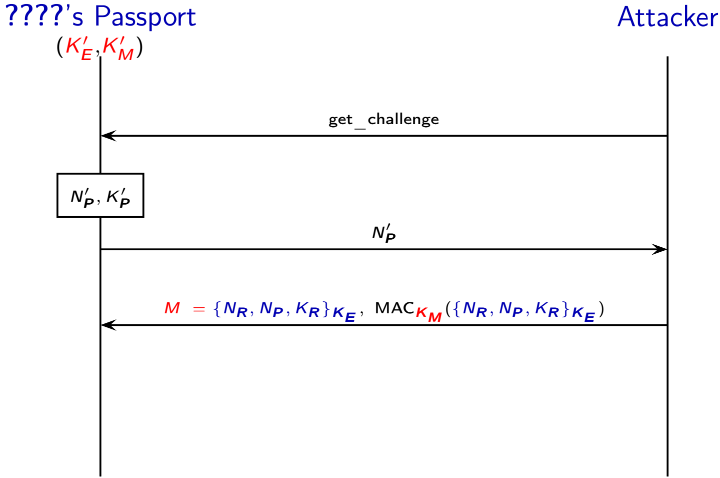
Part 1 of the attack. The attacker eavesdropes on Alice using her passport and records message M.



An attack on the French passport

Part 2 of the attack.

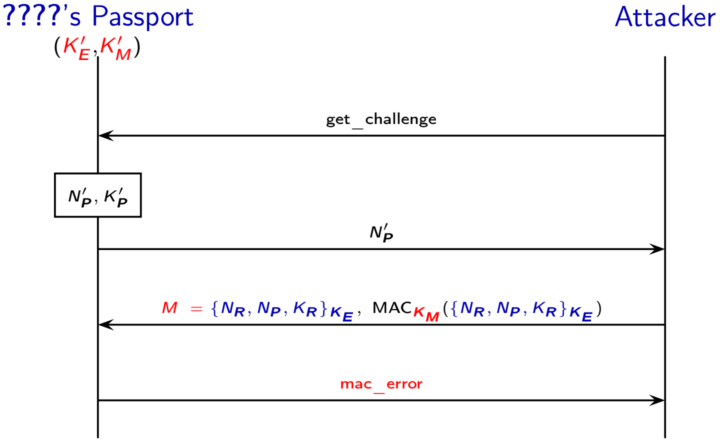
The attacker replays the message M and checks the error code he receives.



An attack on the French passport

Part 2 of the attack.

The attacker replays the message M and checks the error code he receives.

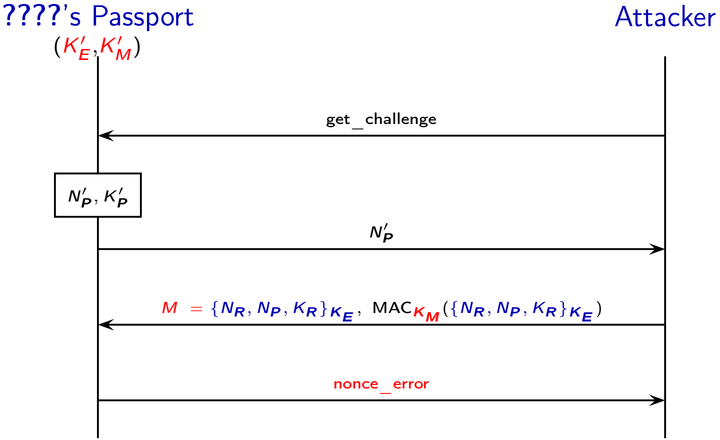


$\Longrightarrow$ MAC check failed $\Longrightarrow$ $K'_M \neq K_M$ $\Longrightarrow$???? is not Alice

An attack on the French passport

Part 2 of the attack.

The attacker replays the message M and checks the error code he receives.



$\Longrightarrow$ MAC check succeeded $\Longrightarrow$ $K'_M = K_M$ $\Longrightarrow$???? is Alice

An attack on the French passport

Attack !

The equivalence does not hold: $P_{\text{same}} \not\approx P_{\text{diff}}$.

More formally,

$$P_{\text{same}} \stackrel{\text{def}}{=} !\mathit{new}\ ke.\mathit{new}\ km.(!P_{\text{BAC}} \mid !R_{\text{BAC}})$$

$$\not\approx$$

$$P_{\text{diff}} \stackrel{\text{def}}{=} !\mathit{new}\ ke.\mathit{new}\ km.(\ P_{\text{BAC}} \mid !R_{\text{BAC}})$$

An attack on the French passport

Attack !

The equivalence does not hold: $P_{\text{same}} \not\approx P_{\text{diff}}$.

More formally,

$$P_{\text{same}} \stackrel{\text{def}}{=} !\textit{new ke}.\textit{new km}.(!P_{\text{BAC}} \mid !R_{\text{BAC}})$$

$$\not\approx$$

$$P_{\text{diff}} \stackrel{\text{def}}{=} !\textit{new ke}.\textit{new km}.(\ P_{\text{BAC}} \mid !R_{\text{BAC}})$$

Exercise: Exhibit the process A that witnesses the fact that these two processes are not in testing equivalence.

Attack !

The equivalence does not hold: $P_{\text{same}} \not\approx P_{\text{diff}}$.

More formally,

$$P_{\text{same}} \stackrel{\text{def}}{=} !\mathit{new}\ ke.\mathit{new}\ km.(!P_{\text{BAC}} \mid !R_{\text{BAC}})$$
$$\not\approx$$
$$P_{\text{diff}} \stackrel{\text{def}}{=} !\mathit{new}\ ke.\mathit{new}\ km.(\ P_{\text{BAC}} \mid !R_{\text{BAC}})$$

Exercise: Exhibit the process A that witnesses the fact that these two processes are not in testing equivalence.

$\longrightarrow A = \mathrm{in}(c, x).\mathrm{out}(c, x).\mathrm{in}(c, y).\mathrm{if}\ y = \text{nonce\_error then out}(ok, \_)$

Some other equivalence-based security properties

The notion of testing equivalence can be used to express:

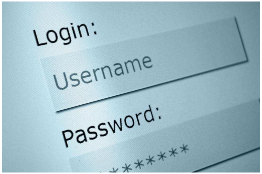
Vote privacy

the fact that a particular voted in a particular way is not revealed to anyone

Strong secrecy

the fact that an adversary cannot see any difference when the value of the secret changes

$\longrightarrow$ stronger than the notion of secrecy as non-deducibility.



Guessing attack

the fact that an adversary can not learn the value of passwords even if he knows that they have been choosen in a particular dictionary.

State of the art in a nutshell (1/2)

for analysing equivalence-based security properties
for an unbounded number of sessions

State of the art in a nutshell (1/2)

for analysing equivalence-based security properties
for an unbounded number of sessions

- undecidable in general even for some fragment for which confidentiality is decidable [Chrétien, Cortier & D., 13]
- some recent decidability results for some restricted fragment *e.g.* tagged protocol, no nonces, a particular set of primitives ... [Chrétien, Cortier & D., Icalp'13, Concur'14, CSF'15]
- ProVerif: a tool that does not correspond to any decidability result for analysing the notion of diff-equivalence (too strong) [Blanchet, Abadi & Fournet, 05]

None of these results is suitable to to analyse vote-privacy, or unlinkability of the BAC protocol.

State of the art in a nutshell (2/2)

for analysing equivalence-based security properties
for a bounded number of sessions

State of the art in a nutshell (2/2)

for analysing equivalence-based security properties
for a bounded number of sessions

A "recent" result [Cheval, Comon & D., 11]

A procedure for deciding testing equivalence for a large class of processes for a bounded number of sessions.

Our class of processes:

- + non-trivial else branches, private channels, and non-deterministic choice;
- – a fixed set of cryptographic primitives (signature, encryption, hash function, mac).

Similar results (for different classes of processes) have been obtained by [Baudet, 05], [Dawson& Tiu, 10], [Chevalier & Rusinowitch, 10], . . .

Privacy using the constraint solving approach

Two main steps:

1. A symbolic exploration of all the possible traces

   The infinite number of possible traces (*i.e.* experiment) are represented by a finite set of constraint systems

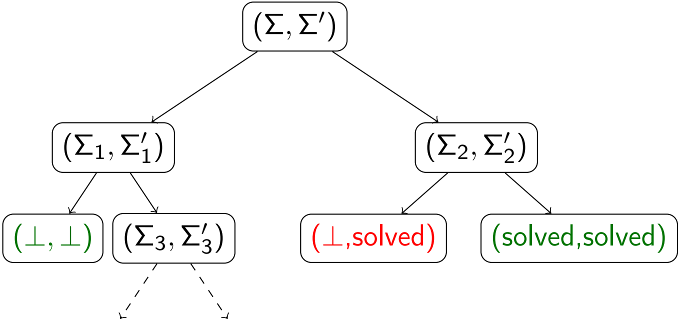
   $\longrightarrow$ this set can be huge (exponential on the number of sessions) !

2. A decision procedure for deciding (symbolic) equivalence between sets of constraint systems

   $\longrightarrow$ this algorithm works quite well

Deciding symbolic equivalence

Main idea: We rewrite pairs (Σ, Σ') of sets of constraint systems (extended to keep track of some information) until a trivial failure or a trivial success is found.



Results on the simplification rules

Termination

Applying blindly the simplification rules does not terminate but there is a particular strategy $\mathcal{S}$ that allows us to ensure termination.

Soundness/Completeness

Let (Σ_0, Σ_0') be pair of sets of constraint systems, and consider a binary tree obtained by applying our simplification rule following a strategy $\mathcal{S}$.

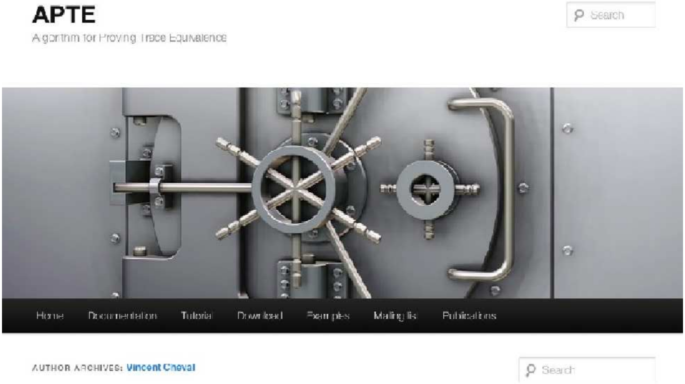
1. soundness: If all leaves of the tree are labeled with $(\perp, \perp)$ or $(\mathit{solved}, \mathit{solved})$, then $\Sigma_0 \approx_s \Sigma_0'$.
2. completeness: if $\Sigma_0 \approx_s \Sigma_0'$, then all leaves of the tree are labeled with $(\perp, \perp)$ or $(\mathit{solved}, \mathit{solved})$.

Theorem

Deciding testing equivalence between processes without replication for classical primitives is decidable.

`http://projects.lsv.ens-cachan.fr/APTE` (Ocaml - 12 KLocs)

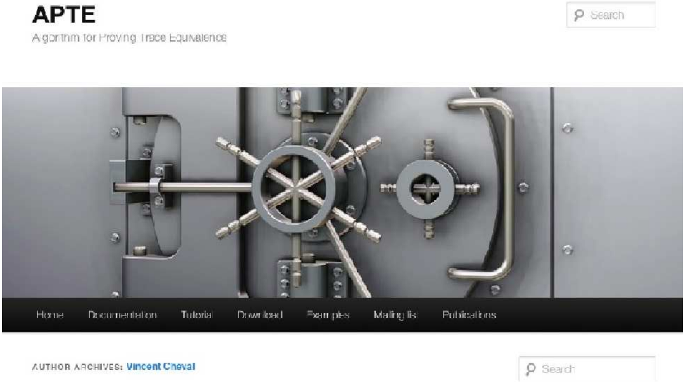
$\longrightarrow$ developed by Vincent Cheval [Cheval, TACAS'14]



APTE- Algorithm for Proving Testing Equivalence

`http://projects.lsv.ens-cachan.fr/APTE` (Ocaml - 12 KLocs)

$\longrightarrow$ developed by Vincent Cheval [Cheval, TACAS'14]



$\longrightarrow$ but a limited practical impact because it scales badly

Partial order reduction for security protocols

part of the PhD thesis of L. Hirschi

Main objective

to develop POR techniques that are suitable for analysing security protocols (especially testing equivalence)

Partial order reduction for security protocols

part of the PhD thesis of L. Hirschi

Main objective

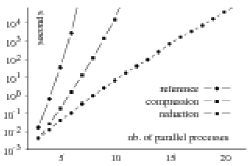
to develop POR techniques that are suitable for analysing security protocols (especially testing equivalence)

Example: $\mathrm{in}(c_1, x_1).\mathrm{out}(c_1, \mathrm{ok}) \mid \mathrm{in}(c_2, x_2).\mathrm{out}(c_2, \mathrm{ok})$

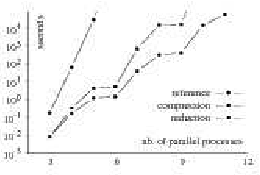
We propose two optimizations:

1. compression: we impose a simple strategy on the exploration of the available actions (roughly outputs are performed first and using a fixed arbitrary order)
2. reduction: we avoid exploring some redundant traces taking into account the data that are exchanged

Practical impact of our optimizations (in APTE)

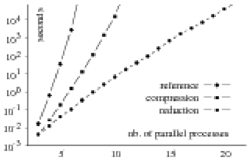


Toy example

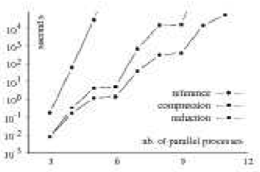


Denning Sacco protocol

$\longrightarrow$ Each optimisation brings an exponential speedup.



Toy example



Denning Sacco protocol

$\longrightarrow$ Each optimisation brings an exponential speedup.

Protocol	reference	with POR
Yahalom (3-party)	4	5
Needham Schroeder (3-party)	4	7
Private Authentication (2-party)	4	7
E-Passport PA (2-party)	4	9
Denning-Sacco (3-party)	5	10
Wide Mouthed Frog (3-party)	6	13

Maximum number of parallel processes verifiable in 20 hours.

$\longrightarrow$ Our optimisations make Apte much more useful in practice for investigating interesting scenarios.

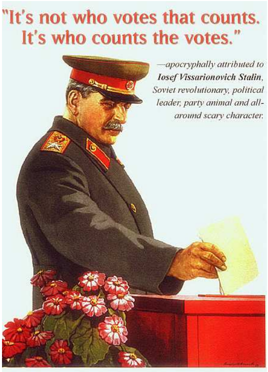
Electronic voting

Electronic voting

Elections are a security-sensitive process which is the cornerstone of modern democracy

Advantages:

- convenient (you can vote from home)
- efficient for recording and tallying



... but risk of large scale, undetected fraud !

$\longrightarrow$ Our goal: a precise modelling of protocols and security properties which allow a rigorous analysis, and to explicit trust assumptions.

A variety of security properties

Eligibility: only legitimate voters can vote, and only once

No early results: no early results can be obtained which could influence the remaining voters

Vote-privacy/Receipt-freeness/Coercion-resistance: the fact that a particular voted in a particular way is not revealed to anyone

Individual/Universal verifiability:
a voter can verify that her vote was really counted, and that the published outcome is the sum of all the votes

A variety of security properties

Individual verifiability

Fairness

Receipt freeness

Coercion resistance

Universal verifiability

Eligibility

Vote privacy

A variety of security properties

Individual verifiability

Fairness

Receipt freeness

Coercion resistance

Universal verifiability

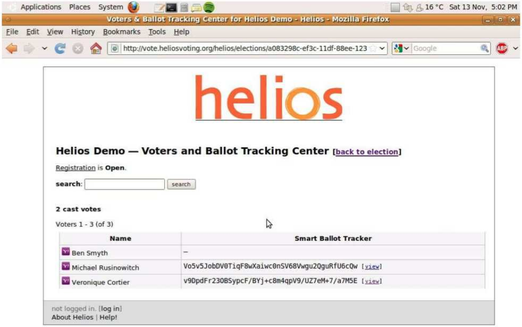
Eligibility

Vote privacy

$\longrightarrow$ e-voting protocols are often complex, rely on non classical cryptographic primitives (*e.g.* blind signature, homomorphic encryption), and only satisfy a subset of the security properties mentioned above.

Helios

⟶ developed by Ben Adida *et al.*



⟶ already in use: election at UCL (Belgium) and Princeton university, election of the IACR board (major association in cryptography), ...

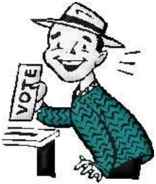
`https://vote.heliosvoting.org`

Behavior of Helios (simplified)

Voting phase: vote 0 or 1 using randomized encryption

Bulletin board

Alice	$\{v_A\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{v_B\}^{r_B}_{\mathrm{pub}(S)}$
Chris	$\{v_C\}^{r_C}_{\mathrm{pub}(S)}$



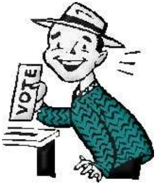
Behavior of Helios (simplified)

Voting phase: vote 0 or 1 using randomized encryption

Bulletin board

Alice	$\{v_A\}^{r_A}_{\text{pub}(S)}$
Bob	$\{v_B\}^{r_B}_{\text{pub}(S)}$
Chris	$\{v_C\}^{r_C}_{\text{pub}(S)}$

$\xleftarrow{\{v_D\}^{r_D}\text{pub}(S)}$



Behavior of Helios (simplified)

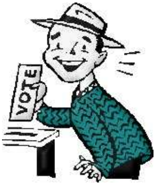
Voting phase: vote 0 or 1 using randomized encryption

Bulletin board	
Alice	$\{v_A\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{v_B\}^{r_B}_{\mathrm{pub}(S)}$
Chris	$\{v_C\}^{r_C}_{\mathrm{pub}(S)}$
David	$\{v_D\}^{r_D}_{\mathrm{pub}(S)}$

Behavior of Helios (simplified)

Voting phase: vote 0 or 1 using randomized encryption

Bulletin board	
Alice	$\{v_A\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{v_B\}^{r_B}_{\mathrm{pub}(S)}$
Chris	$\{v_C\}^{r_C}_{\mathrm{pub}(S)}$
David	$\{v_D\}^{r_D}_{\mathrm{pub}(S)}$



Tallying phase: using homomorphic encryption

$$\{v_A\}^{r_A}_{\mathrm{pub}(S)} \times \{v_B\}^{r_B}_{\mathrm{pub}(S)} \times \ldots = \{v_A + v_B + \ldots\}^{f(r_A, r_B, \ldots)}_{\mathrm{pub}(S)}$$

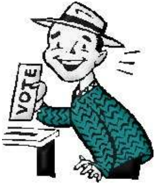
$\longrightarrow$ Only the final result needs to be decrypted !

Behavior of Helios (simplified)

Voting phase: vote 0 or 1 using randomized encryption

Bulletin board

Alice	$\{v_A\}_{\mathrm{pub}(S)}^{r_A}$
Bob	$\{v_B\}_{\mathrm{pub}(S)}^{r_B}$
Chris	$\{v_C\}_{\mathrm{pub}(S)}^{r_C}$
David	$\{v_D\}_{\mathrm{pub}(S)}^{r_D}$



Tallying phase: using homomorphic encryption

$$\{v_A\}_{\mathrm{pub}(S)}^{r_A} \times \{v_B\}_{\mathrm{pub}(S)}^{r_B} \times \ldots = \{v_A + v_B + \ldots\}_{\mathrm{pub}(S)}^{f(r_A, r_B, \ldots)}$$

$\longrightarrow$ Only the final result needs to be decrypted !

A malicious voter can cheat !

Behavior of Helios (simplified)

Voting phase: vote 0 or 1 using randomized encryption

Bulletin board	
Alice	$\{v_A\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{v_B\}^{r_B}_{\mathrm{pub}(S)}$
Chris	$\{v_C\}^{r_C}_{\mathrm{pub}(S)}$
David	$\{v_D\}^{r_D}_{\mathrm{pub}(S)}$

Tallying phase: using homomorphic encryption

$$\{v_A\}^{r_A}_{\mathrm{pub}(S)} \times \{v_B\}^{r_B}_{\mathrm{pub}(S)} \times \ldots = \{v_A + v_B + \ldots\}^{f(r_A, r_B, \ldots)}_{\mathrm{pub}(S)}$$

$\longrightarrow$ Only the final result needs to be decrypted !

A malicious voter can cheat !

$$\{v_D\}_{\mathrm{pub}(S)} \text{ "+" proof of knowledge that } v_D \text{ is equal to } 0 \text{ or } 1$$

Modelling vote-privacy

Classically anonymity properties are modeled using testing equivalences between two slightly different processes, but

- changing the identity does not work, as identities are revealed
- changing the vote does not work, as the votes are revealed at the end
- a correct protocol respecting privacy may in some situation reveal how a participant voted: the case of unanimity

Modelling vote-privacy

Classically anonymity properties are modeled using testing equivalences between two slightly different processes, but

- changing the identity does not work, as identities are revealed
- changing the vote does not work, as the votes are revealed at the end
- a correct protocol respecting privacy may in some situation reveal how a participant voted: the case of unanimity

Vote privacy [Kremer and Ryan, 2005]

$$S[V_A(yes) \mid V_B(no)] \approx_t S[V_A(no) \mid V_B(yes)]$$

A votes yes B votes no	A votes no B votes yes

Helios

Individual and universal verifiability

Helios satisfies a priori the verifiability properties.

Helios

Individual and universal verifiability

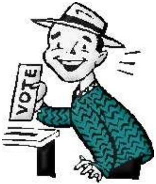
Helios satisfies a priori the verifiability properties.

Vote-privacy, receipt-freeness, coercion resistance

- Helios has not beed designed to satisfy receipt-freeness and coercion-resistance
  $\longrightarrow$ it is possible to obtain a receipt of his vote, namely (v_D, r_D).

Bulletin board	
	$\ldots$
David	$\{v_D\}^{r_D}_{\mathrm{pub}(S)}$

$\xleftarrow{\{v_D\}^{r_D}_{\mathrm{pub}(S)}}$



Individual and universal verifiability

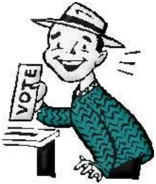
Helios satisfies a priori the verifiability properties.

Vote-privacy, receipt-freeness, coercion resistance

- Helios has not beed designed to satisfy receipt-freeness and coercion-resistance
  $\longrightarrow$ it is possible to obtain a receipt of his vote, namely (v_D, r_D).

Bulletin board	
	...
David	$\{v_D\}^{r_D}_{\mathrm{pub}(S)}$

$\xleftarrow{\{v_D\}^{r_D}_{\mathrm{pub}(S)}}$



- **Helios does not satisfy vote-privacy !**

[Cortier & Smyth, 11]

Vote-privacy in Helios

$$S[V_A(\textit{yes})|\ V_B(\textit{no})] \approx_t S[V_A(\textit{no})|\ V_B(\textit{yes})]$$

A votes yes B votes no	A votes no B votes yes

Description of the attack:

Bulletin board

Alice	$\{\textit{yes}\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{\textit{no}\}^{r_B}_{\mathrm{pub}(S)}$

Bulletin board

Alice	$\{\textit{no}\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{\textit{yes}\}^{r_B}_{\mathrm{pub}(S)}$

Vote-privacy in Helios

$$S[V_A(\textit{yes}) \mid V_B(\textit{no})] \approx_t S[V_A(\textit{no}) \mid V_B(\textit{yes})]$$

A votes yes B votes no	A votes no B votes yes

Description of the attack:

Bulletin board	
Alice	$\{\textit{yes}\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{\textit{no}\}^{r_B}_{\mathrm{pub}(S)}$
Charlie	

Bulletin board	
Alice	$\{\textit{no}\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{\textit{yes}\}^{r_B}_{\mathrm{pub}(S)}$
Charlie	

$\longrightarrow$ Charlies simply copies Alice's vote !

Vote-privacy in Helios

$$S[V_A(yes)|\ V_B(no)] \approx_t S[V_A(no)|\ V_B(yes)]$$

A votes yes B votes no	A votes no B votes yes

Description of the attack:

Bulletin board	
Alice	$\{yes\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{no\}^{r_B}_{\mathrm{pub}(S)}$
Charlie	$\{yes\}^{r_A}_{\mathrm{pub}(S)}$

Bulletin board	
Alice	$\{no\}^{r_A}_{\mathrm{pub}(S)}$
Bob	$\{yes\}^{r_B}_{\mathrm{pub}(S)}$
Charlie	$\{no\}^{r_A}_{\mathrm{pub}(S)}$

$\longrightarrow$ Charlies simply copies Alice's vote !

Video of the attack at `http://www.youtube.com/watch?v=fWvl9uJgpc0`

In conclusion

(few words)

Limitations of the symbolic approach

1. the algebraic properties of the primitives are abstracted away
   $\longrightarrow$ no guarantee if the protocol relies on an encryption that satisfies some additional properties (*e.g.* RSA, ElGamal)

2. only the specification is analysed and not the implementation
   $\longrightarrow$ most of the passports are actually linkable by a carefull analysis of time or message length.

   `http://www.loria.fr/~glondu/epassport/attaque-tailles.html`

3. when the analysis is done for a bounded number of sessions, not all scenario are checked
   $\longrightarrow$ no guarantee if the protocol is used one more time !

Conclusion

A need of formal methods in verification of security protocols.
Regarding confidentiality (or authentication), powerful tool support that are nowdays used by industrials and security agencies.

It remains a lot to do for analysing privacy-type properties:

- formal definitions of some sublte security properties;
  $\longrightarrow$ receipt-freeness, coercion-resistance in e-voting
- algorithms (and tools!) for checking automatically testing equivalence for various cryptographic primitives;
  $\longrightarrow$ homomorphic encryption used in e-voting,
- more composition results.
  $\longrightarrow$ Could we derive some security guarantees of the whole e-passport application from the analysis performed on each subprotocol in isolation?