Analysing privacy-type properties in cryptographic protocols

Stéphanie Delaune

LSV, CNRS & ENS Cachan, France

Thursday, April 9th, 2015

Cryptographic protocols everywhere !

Cryptographic protocols

- small programs designed to secure communication (*e.g.* secrecy, authentication, anonymity, . . .)
- use cryptographic primitives (*e.g.* encryption, signature, ...)

The network is unsecure!

Communications take place over a public network like the Internet.

Cryptographic protocols everywhere !

Cryptographic protocols

- small programs designed to secure communication (*e.g.* secrecy, authentication, anonymity, . . .)
- use cryptographic primitives (*e.g.* encryption, signature, ...)

It becomes more and more important to protect our privacy.

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

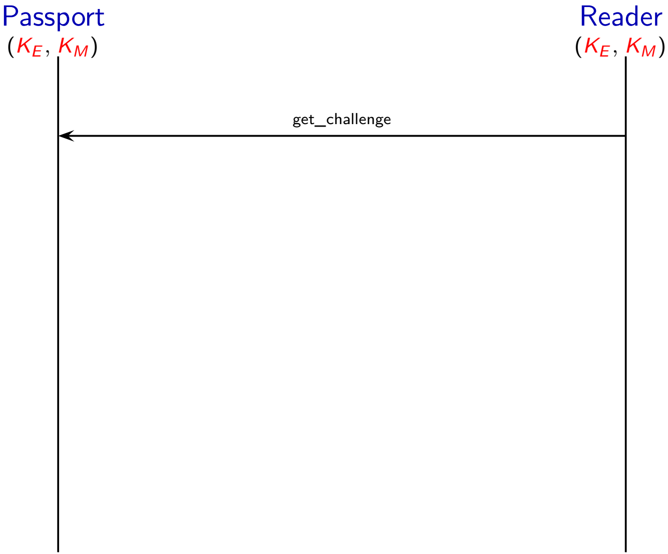
Unlinkability aims to ensure *that a user may make multiple uses of a service or resource without others being able to link these uses together.*

Basic Acccess Control (BAC) protocol

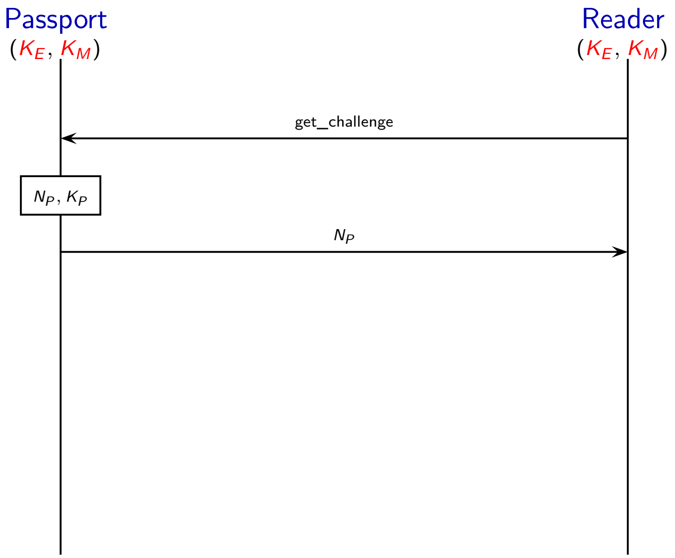
Passport
(K_E, K_M)

Reader
(K_E, K_M)

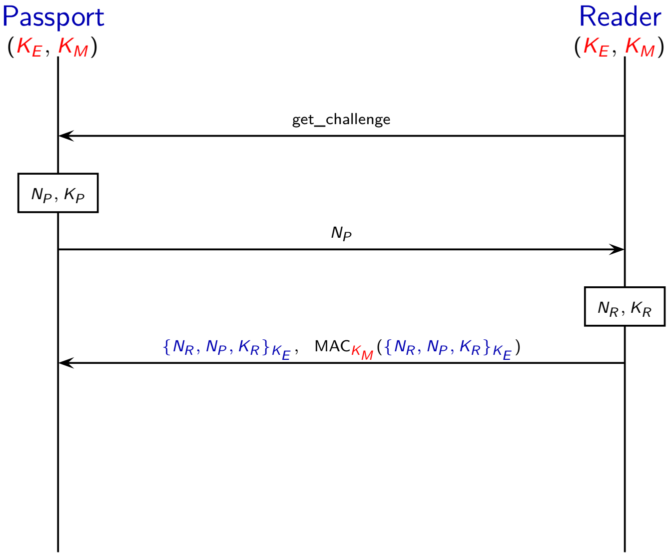
Basic Acccess Control (BAC) protocol



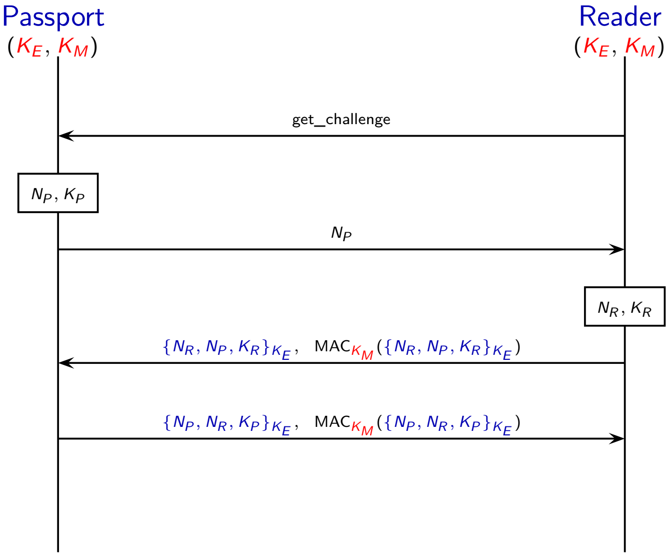
Basic Acccess Control (BAC) protocol



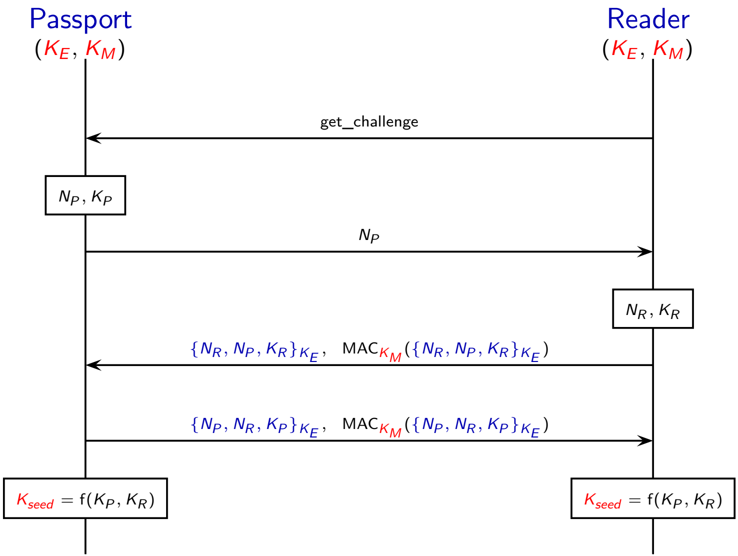
Basic Acccess Control (BAC) protocol



Basic Acccess Control (BAC) protocol



Basic Acccess Control (BAC) protocol



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,

$$!\text{new } ke.\text{new } km.(\underbrace{!P_{\text{BAC}}}_{\uparrow} \mid !R_{\text{BAC}}) \overset{?}{\approx} !\text{new } ke.\text{new } km.(\underbrace{P_{\text{BAC}}}_{\uparrow} \mid !R_{\text{BAC}})$$

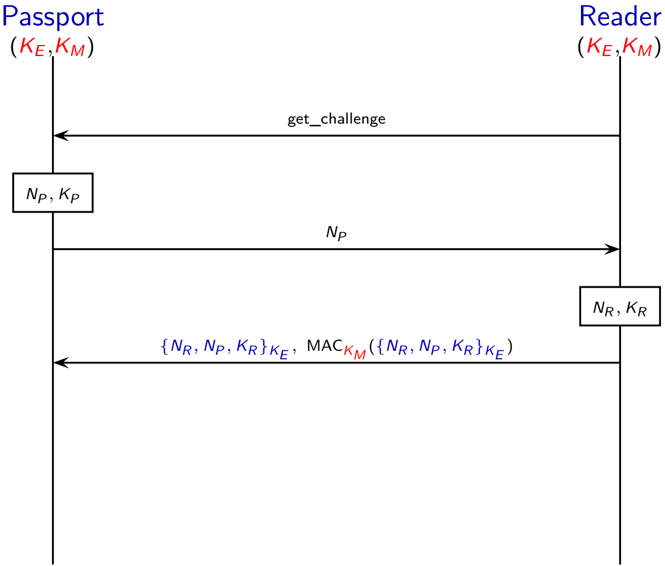
many sessions for each passport

only one session for each passport

(we still have to formalize the processes and the notion of equivalence)

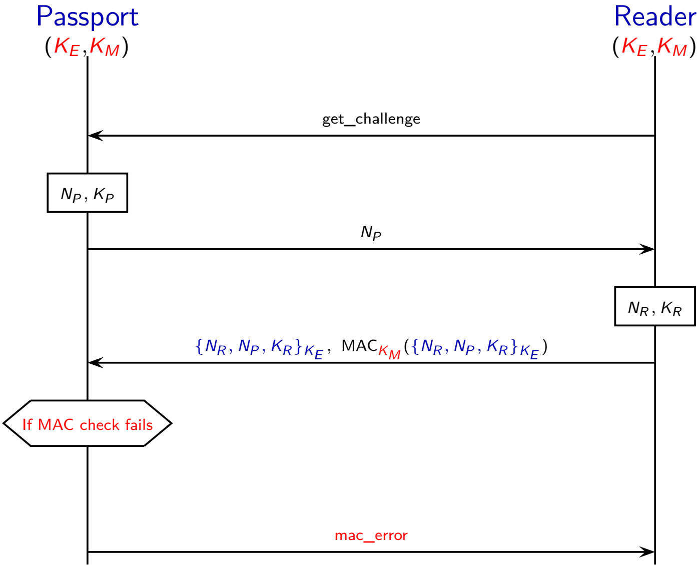
French electronic passport

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



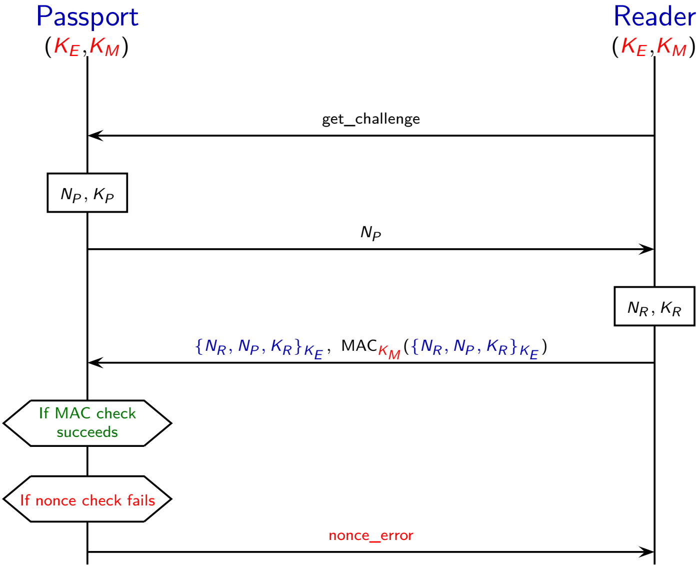
French electronic passport

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



French electronic passport

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



An attack on the French passport [Chothia & Smirnov, 10]

Attack against unlinkability

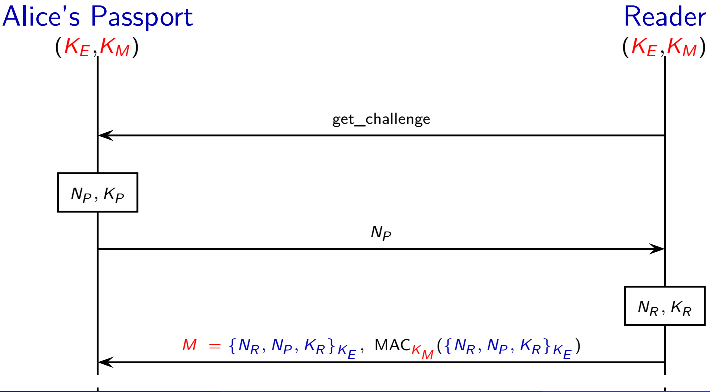
An attacker can track a French passport, provided he has once witnessed a successful authentication.

An attack on the French passport [Chothia & Smirnov, 10]

Attack against unlinkability

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

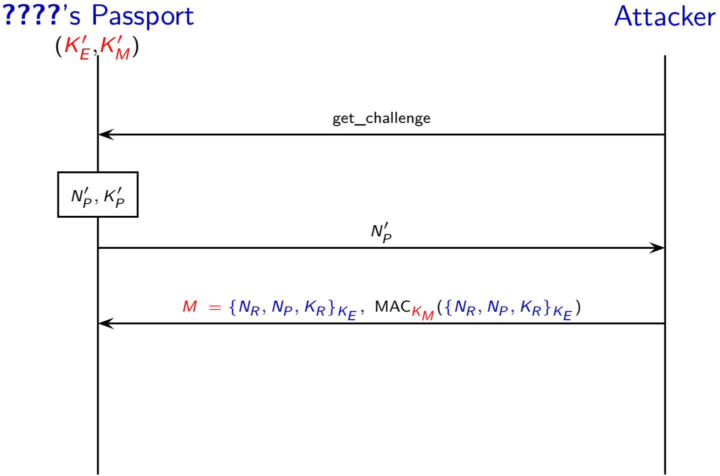
Part 1 of the attack. The attacker eavesdropes on Alice using her passport and records message M.



An attack on the French passport [Chothia & Smirnov, 10]

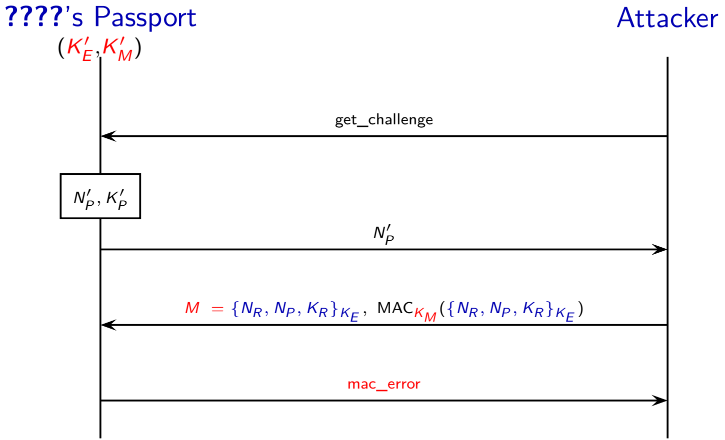
Part 2 of the attack.

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



Part 2 of the attack.

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

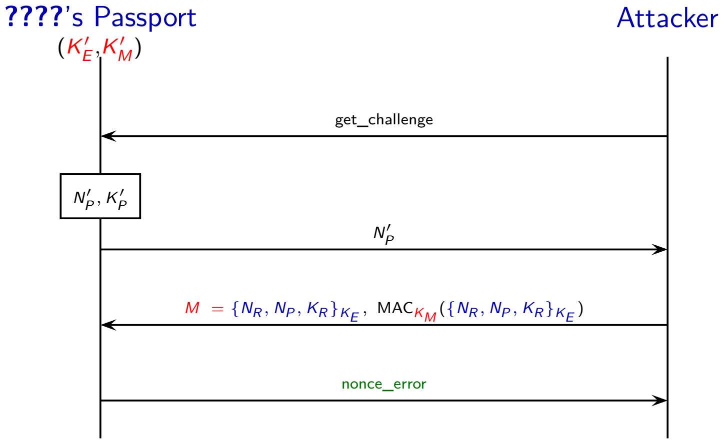


$\Longrightarrow$ MAC check failed $\Longrightarrow$ $K'_M \neq K_M$ $\Longrightarrow$ **????** is not Alice

An attack on the French passport [Chothia & Smirnov, 10]

Part 2 of the attack.

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

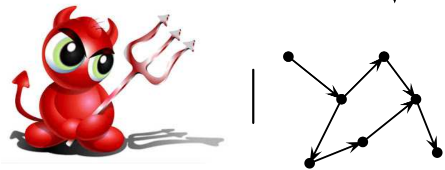


$\Longrightarrow$ MAC check succeeded $\Longrightarrow$ $K'_M = K_M$ $\Longrightarrow$ **????** is Alice

Outline

Does the protocol *satisfy* a security property?

Modelling



$\models \varphi$

Outline of the remaining of this talk

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

$\longrightarrow$ we focus here on privacy-type security properties

Part I

Modelling cryptographic protocols

and their security properties

Protocols as processes

Applied pi calculus [Abadi & Fournet, 01]

basic programming language with constructs for concurrency and communication

$\longrightarrow$ based on the π-calculus [Milner *et al.*, 92] ...

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

Protocols as processes

Applied pi calculus [Abadi & Fournet, 01]

basic programming language with constructs for concurrency and communication

$\longrightarrow$ based on the π-calculus [Milner *et al.*, 92] ...

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

... but messages that are exchanged are not necessarily atomic !

Messages as terms

Messages are abstracted by (ground) terms

Ground terms are built over a set of names $\mathcal{N}$, and a signature $\mathcal{F}$.

t	$::=$	n	name n
	$\mid$	$f(t_1, \ldots, t_k)$	application of symbol $f \in \mathcal{F}$

Messages as terms

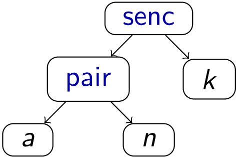
Messages are abstracted by (ground) terms

Ground terms are built over a set of names $\mathcal{N}$, and a signature $\mathcal{F}$.

$$
\begin{array}{rcll}
\mathrm{t} & ::= & n & \text{name } n \\
& | & f(t_1, \ldots, t_k) & \text{application of symbol } f \in \mathcal{F}
\end{array}
$$

Example: representation of $\{a, n\}_k$

- Names: n, k, a
- constructors: senc, pair,



Messages as terms

Messages are abstracted by (ground) terms

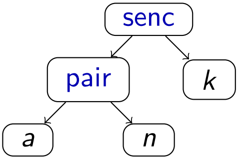
Ground terms are built over a set of names $\mathcal{N}$, and a signature $\mathcal{F}$.

$$
\begin{array}{rcll}
t & ::= & n & \text{name } n \\
 & | & f(t_1, \ldots, t_k) & \text{application of symbol } f \in \mathcal{F}
\end{array}
$$

$\longrightarrow$ The term algebra is equipped with an equational theory E.

Example: representation of $\{a, n\}_k$

- Names: n, k, a
- constructors: senc, pair,
- destructors: sdec, proj_1, proj_2.



$\longrightarrow$ $\text{sdec}(\text{senc}(x, y), y) = x$, $\text{proj}_1(\text{pair}(x, y)) = x$, $\text{proj}_2(\text{pair}(x, y)) = y$.

Going back to the e-passport

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

Going back to the e-passport

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

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

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

Semantics

Semantics $\rightarrow$:

Comm	$\mathsf{out}(c,u).P \mid \mathsf{in}(c,x).Q \rightarrow P \mid Q\{u/x\}$
Then	$\text{if } u = v \text{ then } P \text{ else } Q \rightarrow P \text{ when } u =_{\mathsf{E}} v$
Else	$\text{if } u = v \text{ then } P \text{ else } Q \rightarrow Q \quad \text{when } u \neq_{\mathsf{E}} v$

Semantics

Semantics $\rightarrow$:

COMM	$\mathsf{out}(c,u).P \mid \mathsf{in}(c,x).Q \rightarrow P \mid Q\{u/x\}$
THEN	if $u = v$ then P else $Q \rightarrow P$ when $u =_{\mathsf{E}} v$
ELSE	if $u = v$ then P else $Q \rightarrow Q$ when $u \neq_{\mathsf{E}} v$

closed by

- structural equivalence ($\equiv$):

$$P \mid Q \equiv Q \mid P, \quad P \mid 0 \equiv P, \quad \ldots$$

- application of evaluation contexts:

$$\frac{P \rightarrow P'}{\mathsf{new}\,n.\,P \rightarrow \mathsf{new}\,n.\,P'} \qquad \frac{P \rightarrow P'}{P \mid Q \rightarrow P' \mid Q}$$

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

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

for all processes A, we have that:

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

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

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

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

for all processes A, we have that:

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

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

Example 1: $\mathsf{out}(a, s) \overset{?}{\approx}_t \mathsf{out}(a, s')$

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

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

for all processes A, we have that:

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

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

Example 1: $\mathsf{out}(a, s) \not\approx_t \mathsf{out}(a, s')$

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

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

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

for all processes A, we have that:

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

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

Example 2:

$$\begin{gathered}\mathsf{new}\ s.\mathsf{out}(a, \mathsf{senc}(s, k)).\mathsf{out}(a, \mathsf{senc}(s, k')) \\ \stackrel{?}{\approx}_t \\ \mathsf{new}\ s, s'.\mathsf{out}(a, \mathsf{senc}(s, k)).\mathsf{out}(a, \mathsf{senc}(s', k'))\end{gathered}$$

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

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

for all processes A, we have that:

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

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

Example 2:

$$\text{new } s.\text{out}(a, \text{senc}(s, k)).\text{out}(a, \text{senc}(s, k'))$$
$$\not\approx_t$$
$$\text{new } s, s'.\text{out}(a, \text{senc}(s, k)).\text{out}(a, \text{senc}(s', k'))$$

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

Security properties - privacy

Privacy-type properties are modelled as equivalence-based properties

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

for all processes A, we have that:

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

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

Question: Are the two following processes in testing equivalence?

$$\mathsf{new}\, s.\mathsf{out}(a, s) \ \stackrel{?}{\approx}_t \ \mathsf{new}\, k.\mathsf{out}(a, \mathsf{senc}(\mathsf{yes}, k))$$

Some privacy-type properties

Unlinkability [Arapinis et al, 2010]

$$!\mathsf{new}\ ke.\mathsf{new}\ km.(!P_{\mathsf{BAC}} \mid !R_{\mathsf{BAC}}) \approx_t !\mathsf{new}\ ke.\mathsf{new}\ km.(P_{\mathsf{BAC}} \mid !R_{\mathsf{BAC}})$$

many sessions for each passport

only one session for each passport

Some privacy-type properties

Unlinkability [Arapinis et al, 2010]

$$!\text{new } ke.\text{new } km.(!P_{\text{BAC}} \mid !R_{\text{BAC}}) \approx_t !\text{new } ke.\text{new } km.(P_{\text{BAC}} \mid !R_{\text{BAC}})$$

many sessions for each passport

only one session for each passport

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

Part II

Designing verification algorithms
for privacy-type properties

How can we check testing equivalence?

Testing equivalence is **undecidable** in general

A first decidability result [Chrétien, Cortier & D., ICALP'13]

The class $\mathcal{C}_{\mathsf{pp}}$ $\quad ! \,\mathsf{in}(c_1, u_1).\mathsf{out}(c_1, v_1) \mid \ldots \mid ! \,\mathsf{in}(c_n, u_n).\mathsf{out}(c_n, v_n)$

with at most one variable in u_i/v_i

Overview of the approach

Testing equivalence between protocols is characterized in terms of equality of languages of (generalized, real-time) deterministic pushdown automata.

A first decidability result [Chrétien, Cortier & D., ICALP'13]

The class $\mathcal{C}_{\text{pp}}$ $\quad !\,\text{in}(c_1, u_1).\text{out}(c_1, v_1) \mid \ldots \mid !\,\text{in}(c_n, u_n).\text{out}(c_n, v_n)$
with at most one variable in u_i/v_i

Overview of the approach

Testing equivalence between protocols is characterized in terms of equality of languages of (generalized, real-time) deterministic pushdown automata.

We have shown that:

1. $P \approx_t Q \;\Leftrightarrow\; L(\mathcal{A}_P) = L(\mathcal{A}_Q)$; and
2. $L(\mathcal{A}) = L(\mathcal{B}) \;\Leftrightarrow\; P_{\mathcal{A}} \approx_t P_{\mathcal{B}}$.

$\longrightarrow$ checking for equivalence of protocols (in $\mathcal{C}_{\text{pp}}$) is as difficult as checking equivalence of deterministic generalized real-time pushdown automata.

A first decidability result [Chrétien, Cortier & D., ICALP'13]

The class $\mathcal{C}_{\text{pp}}$ $\quad ! \, \text{in}(c_1, u_1).\text{out}(c_1, v_1) \mid \ldots \mid ! \, \text{in}(c_n, u_n).\text{out}(c_n, v_n)$
with at most one variable in u_i/v_i

Overview of the approach

Testing equivalence between protocols is characterized in terms of equality of languages of (generalized, real-time) deterministic pushdown automata.

We have shown that:

1. $P \approx_t Q \quad \Leftrightarrow \quad L(\mathcal{A}_P) = L(\mathcal{A}_Q)$; and
2. $L(\mathcal{A}) = L(\mathcal{B}) \quad \Leftrightarrow \quad P_{\mathcal{A}} \approx_t P_{\mathcal{B}}$.

$\longrightarrow$ checking for equivalence of protocols (in $\mathcal{C}_{\text{pp}}$) is as difficult as checking equivalence of deterministic generalized real-time pushdown automata.

Some recent decidability results for larger class of protocols
$\longrightarrow$ [Chrétien, Cortier, D., CONCUR'14 & CSF'15].

A more pragmatic approach [Blanchet *et al.*, LICS'05]

ProVerif: Automated protocol verifier mainly developed by B. Blanchet.

`http://www.proverif.ens.fr`

Main features:

- processes with replication;
- various cryptographic primitives modeled using equations;
- various security properties: secrecy, authentication, and equivalence-based security properties (namely diff-equivalence);

The tool may not terminate or give false attacks. It works well in practice.

A more pragmatic approach [Blanchet *et al.*, LICS'05]

ProVerif: Automated protocol verifier mainly developed by B. Blanchet.

`http://www.proverif.ens.fr`

Main features:

- processes with replication;
- various cryptographic primitives modeled using equations;
- various security properties: secrecy, authentication, and equivalence-based security properties (namely diff-equivalence);

The tool may not terminate or give false attacks. It works well in practice.

Main issue: diff-equivalence is too strong in many situations. ProVerif is not suitable to to analyse vote-privacy, or unlinkability of the BAC protocol.

Testing equivalence (for processes without replication)

For processes without replication
testing equivalence is decidable
(under some reasonnable assumptions)

Testing equivalence (for processes <u>without</u> replication)

For processes <u>without</u> replication
testing equivalence is decidable
(under some reasonnable assumptions)

Some difficulties

1. We have to consider an infinite number of possible behaviours for the attacker (for all quantification over processes).
2. Once the behavior of the attacker is fixed, we have to decide whether the two sequences of messages that are outputted are indistinguishable or not.

Another decidability result

Cheval, Comon-Lundh & D. — **CCS 2011**

A procedure for deciding testing equivalence for a large class of processes implemented in a tool called APTE

Our class of processes:

- + non-trivial else branches, private channels, and non-deterministic choice;
- – but no replication, and a fixed set of cryptographic primitives (signature, symmetric and asymmetric encryptions, hash function, mac, pairs).

Another decidability result

Cheval, Comon-Lundh & D. CCS 2011

A procedure for deciding testing equivalence for a large class of processes implemented in a tool called APTE

Our class of processes:

- + non-trivial else branches, private channels, and non-deterministic choice;
- – but no replication, and a fixed set of cryptographic primitives (signature, symmetric and asymmetric encryptions, hash function, mac, pairs).

Similar results for restricted class of processes have been obtained in [Baudet, 05], [Dawson & Tiu, 10], [Chevalier & Rusinowitch, 10], [Chadha *et al.*, 12], . . .

The procedure in a nutshell

Two main steps:

1. A symbolic exploration of all the possible traces

   The infinite number of possible concrete traces are represented by a finite set of constraint systems.

   $\longrightarrow$ this set is huge (exponential) !

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

   $\longrightarrow$ this algorithm works quite well

The procedure in a nutshell

Two main steps:

1. A symbolic exploration of all the possible traces

   The infinite number of possible concrete traces are represented by a finite set of constraint systems.

   $\longrightarrow$ this set is huge (exponential) !

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

   $\longrightarrow$ this algorithm works quite well

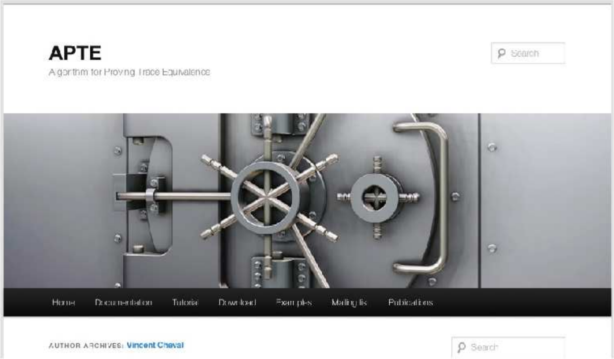
Some applications

- unlinkability in RFID protocols (*e.g.* e-passport protocol)
- anonymity (*e.g.* private authentication protocol)

APTE- Algorithm for Proving Trace Equivalence

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

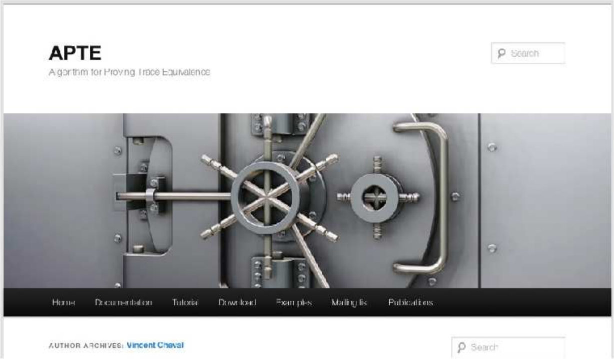
$\longrightarrow$ developed by Vincent Cheval [Cheval, TACAS'14]



APTE- Algorithm for Proving Trace 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

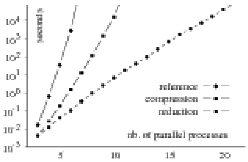
to develop POR techniques that are suitable for analysing security protocols (especially testing equivalence)

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

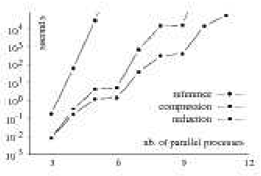
We propose two optimizations:

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 redunant traces taking into account the data that are exchanged

Practical impact of our optimizations (in APTE)



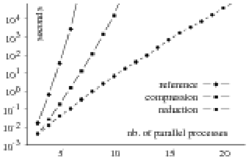
Toy example



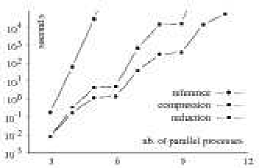
Denning Sacco protocol

$\longrightarrow$ Each optimisation brings an exponential speedup.

Practical impact of our optimizations (in APTE)



Toy example



Denning Sacco protocol

⟶ 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.

⟶ Our optimisations make Apte much more useful in practice for investigating interesting scenarios.

Conclusion

A need of formal methods in verification of security protocols.

Regarding confidentiality (or authentication), powerful tool support are nowdays available and sometimes used by industrials and security agencies.

Conclusion

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

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

- formal definitions of some sublte security properties
- algorithms (and tools!) for checking automatically testing equivalence for various cryptographic primitives;
- more composition results.

VIP - Verification of Indistinguishability Properties

Main topics of the ANR JCJC - VIP project
(Jan. 2012 - Dec 2015)
`http://www.lsv.ens-cachan.fr/Projects/anr-vip/`