A process algebraic analysis of privacy-type properties in cryptographic protocols

Stéphanie Delaune

LSV, CNRS & ENS Cachan, France

Saturday, September 6th, 2014

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!

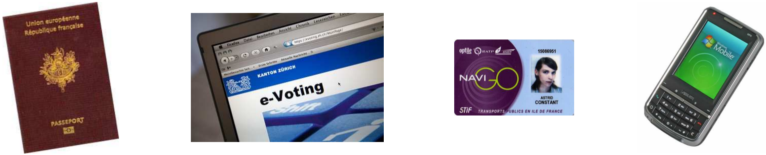
Communications take place over a public network like the Internet.

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

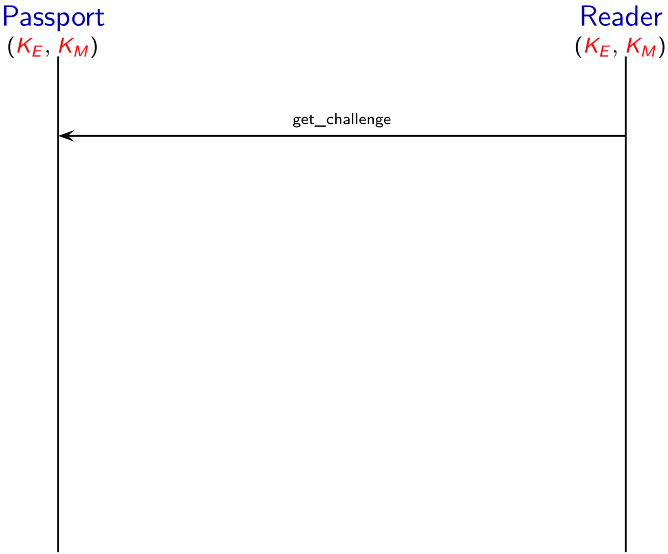
Unlinkability aims to ensure *that a user may make multiple uses of a service or resource without others being able to link these uses together.*

Basic Acccess Control (BAC) protocol

Passport
(K_E, K_M)

Reader
(K_E, K_M)

Basic Acccess Control (BAC) protocol



Passport

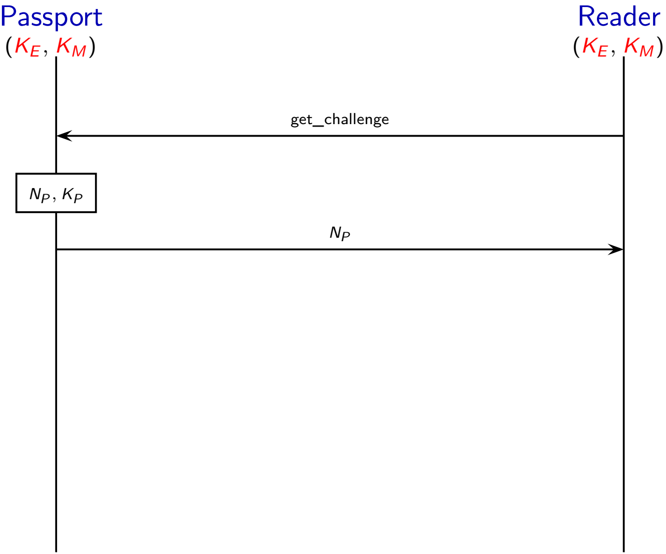
(K_E, K_M)

Reader

(K_E, K_M)

get_challenge

Basic Acccess Control (BAC) protocol



Passport

(K_E, K_M)

Reader

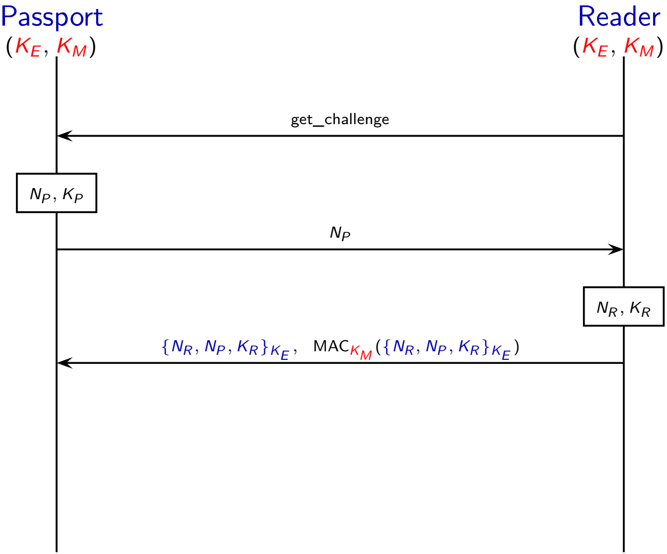
(K_E, K_M)

get_challenge

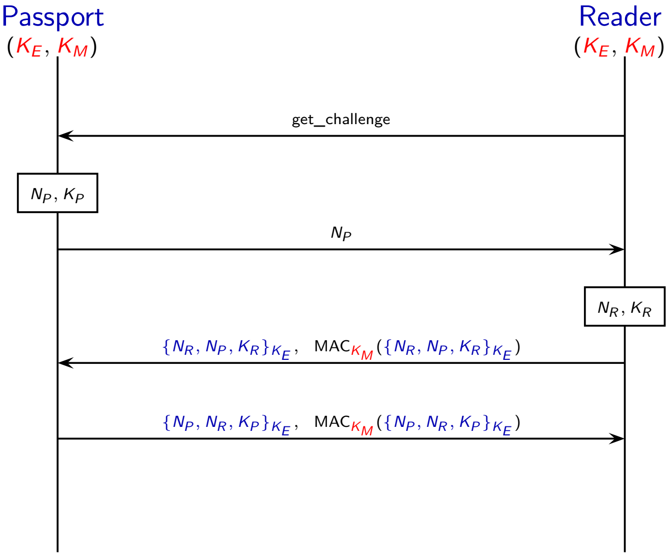
N_P, K_P

N_P

Basic Acccess Control (BAC) protocol



Basic Acccess Control (BAC) protocol



Passport (K_E, K_M)

Reader (K_E, K_M)

get_challenge

N_P, K_P

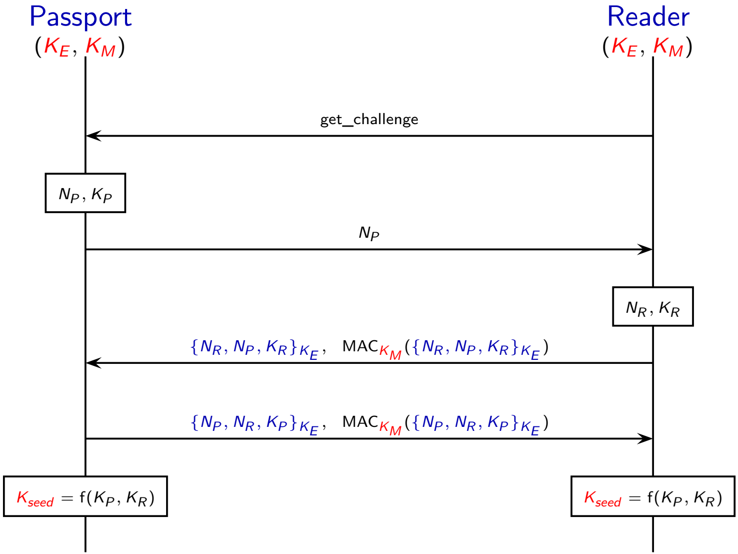
N_P

N_R, K_R

$\{N_R, N_P, K_R\}_{K_E}, \quad \mathrm{MAC}_{K_M}(\{N_R, N_P, K_R\}_{K_E})$

$\{N_P, N_R, K_P\}_{K_E}, \quad \mathrm{MAC}_{K_M}(\{N_P, N_R, K_P\}_{K_E})$

Basic Acccess Control (BAC) protocol



Passport

(K_E, K_M)

Reader

(K_E, K_M)

get_challenge

N_P, K_P

N_P

N_R, K_R

$\{N_R, N_P, K_R\}_{K_E}, \quad \mathrm{MAC}_{K_M}(\{N_R, N_P, K_R\}_{K_E})$

$\{N_P, N_R, K_P\}_{K_E}, \quad \mathrm{MAC}_{K_M}(\{N_P, N_R, K_P\}_{K_E})$

$K_{seed} = \mathrm{f}(K_P, K_R)$

$K_{seed} = \mathrm{f}(K_P, K_R)$

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,

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

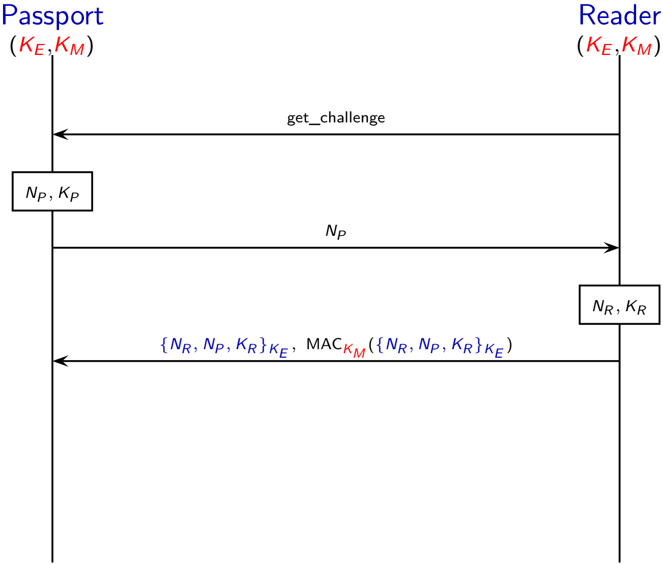
many sessions
for each passport

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.



Passport (K_E, K_M)

Reader (K_E, K_M)

get_challenge

N_P, K_P

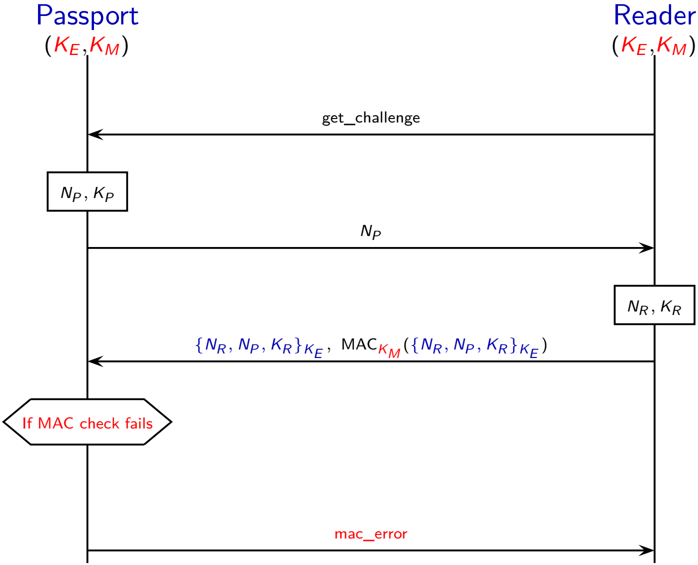
N_P

N_R, K_R

$\{N_R, N_P, K_R\}_{K_E},\ \mathrm{MAC}_{K_M}(\{N_R, N_P, K_R\}_{K_E})$

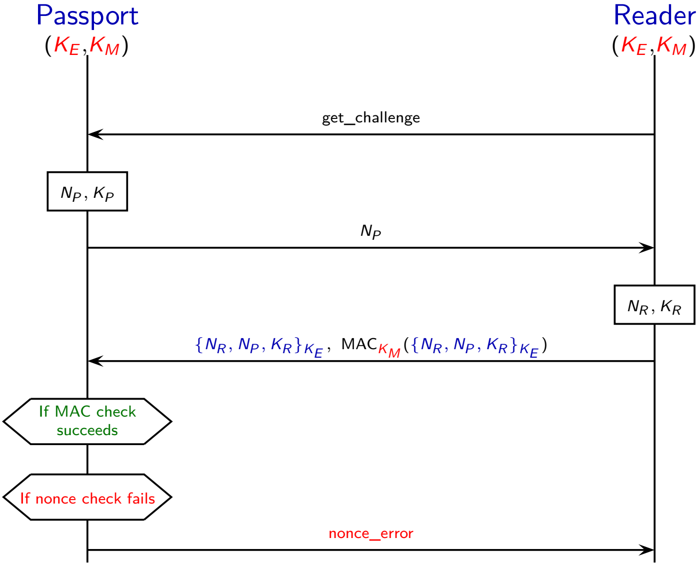
French electronic passport

⟶ the passport must reply to all received messages.



French electronic passport

⟶ the passport must reply to all received messages.



Passport

(K_E, K_M)

Reader

(K_E, K_M)

get_challenge

N_P, K_P

N_P

N_R, K_R

$\{N_R, N_P, K_R\}_{K_E}, \ \mathrm{MAC}_{K_M}(\{N_R, N_P, K_R\}_{K_E})$

If MAC check succeeds

If nonce check fails

nonce_error

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

Attack against unlinkability

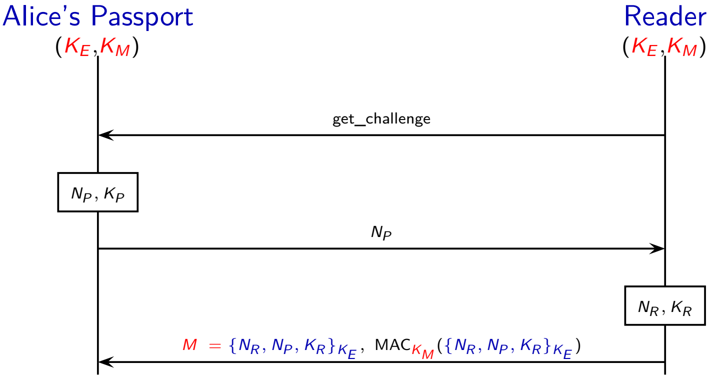
An attacker can track a French passport, provided he has once witnessed a successful authentication.

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.

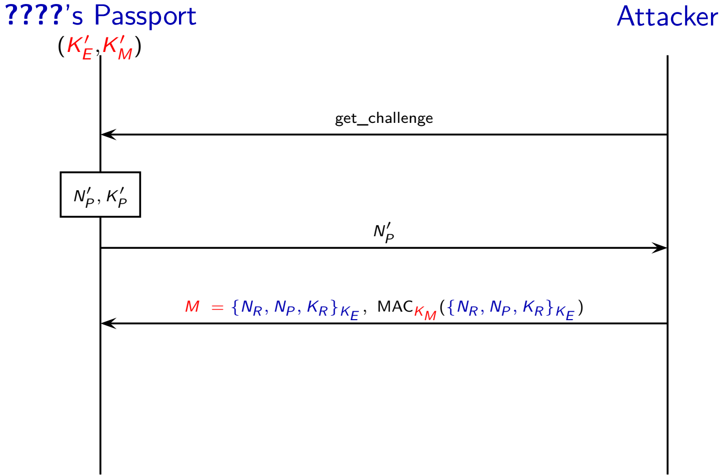
Part 1 of the attack. The attacker eavesdropes on Alice using her passport and records message M.



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.



????'s Passport

(K'_E, K'_M)

Attacker

get_challenge

N'_P, K'_P

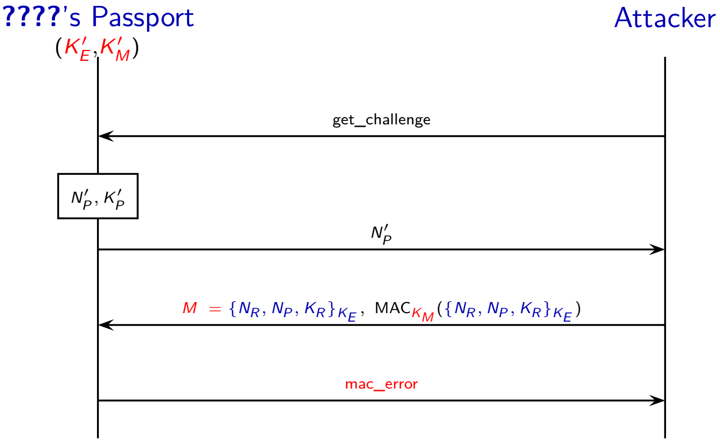
N'_P

$M = \{N_R, N_P, K_R\}_{K_E},\ \mathrm{MAC}_{K_M}(\{N_R, N_P, K_R\}_{K_E})$

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.

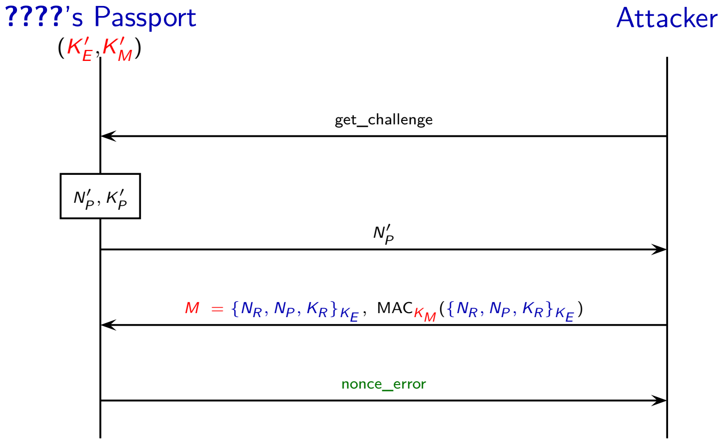


$\Longrightarrow$ MAC check failed $\Longrightarrow$ $K'_M \neq K_M$ $\Longrightarrow$ **????** is not Alice

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.

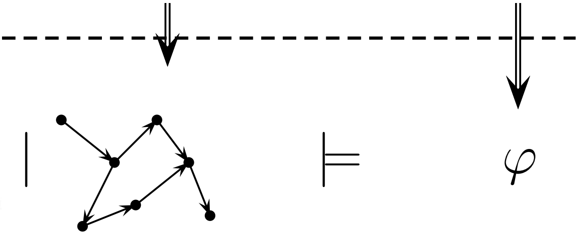


$\Longrightarrow$ MAC check succeeded $\Longrightarrow$ $K'_M = K_M$ $\Longrightarrow$ **????** is Alice

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

⟶ 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

⟶ 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

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 $\mathcal{F}$.

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

Messages as terms

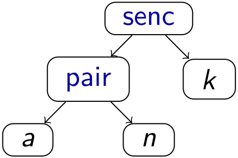
Messages are abstracted by (ground) terms

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

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

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

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



Messages as terms

Messages are abstracted by (ground) terms

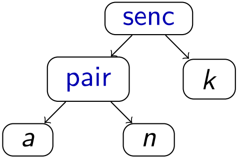
Ground terms are built over a set of names $\mathcal{N}$, and a signature $\mathcal{F}$.

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

⟶ 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.



⟶ $\mathsf{sdec}(\mathsf{senc}(x, y), y) = x, \quad \mathsf{proj}_1(\mathsf{pair}(x, y)) = x, \quad \mathsf{proj}_2(\mathsf{pair}(x, y)) = y.$

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

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

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

Modelling Passport's role

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

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

Semantics

Semantics $\rightarrow$:

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

Semantics

Semantics $\rightarrow$:

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

closed by

- structural equivalence ($\equiv$):

$$P \mid Q \equiv Q \mid P, \qquad P \mid 0 \equiv P, \qquad \dots$$

- 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:

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

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:

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

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

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?

$$\text{new } s.\text{out}(a, s) \ \overset{?}{\approx}_t \ \text{new } s.\text{new } k.\text{out}(a, \text{senc}(s, k))$$

Some privacy-type properties

Unlinkability [Arapinis et al, 2010]

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

many sessions
for each passport

only one session
for each passport

Some privacy-type properties

Unlinkability [Arapinis et al, 2010]

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

many sessions
for each passport

only one session
for each passport

Vote privacy [Kremer and Ryan, 2005]

$$V_A(\mathit{yes}) \approx_t V_A(\mathit{no})$$

Some privacy-type properties

Unlinkability [Arapinis et al, 2010]

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

many sessions
for each passport

only one session
for each passport

Vote privacy [Kremer and Ryan, 2005]

$$V_A(yes) \underset{\uparrow}{\mid} V_B(no) \approx_t V_A(no) \underset{\uparrow}{\mid} V_B(yes)$$

A votes yes
B votes no

A votes no
B votes yes

Some privacy-type properties

Unlinkability [Arapinis et al, 2010]

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

many sessions
for each passport

only one session
for each passport

Vote privacy [Kremer and Ryan, 2005]

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

A votes yes
B votes no

A votes no
B votes yes

$\longrightarrow$ often requires some assumptions $S[\_]$

Part II

Designing verification algorithms for privacy-type properties

Difficulties when checking testing equivalence $P \approx_t Q$

testing equivalence is undecidable in general

Difficulties when checking testing equivalence $P \approx_t Q$

testing equivalence is undecidable in general

Processes without replication

- We still have to consider any possible behavior for the attacker (for all quantification over processes).

  $\longrightarrow$ no hope to test each possible behavior of the attacker in turn

- Once the behavior of the attacker is fixed, we still have to decide whether the two sequences of messages that are outputted are indistinguishable or not.

  $\longrightarrow$ the so-called static equivalence problem.

Static equivalence

Static equivalence $\sigma \sim \sigma'$ (modulo E)

Two sequences of messages $\sigma = \{w_1 \rightarrow u_1, \ldots, w_n \rightarrow u_n\}$ and $\sigma' = \{w_1 \rightarrow u'_1, \ldots, w_n \rightarrow u'_n\}$ are in static equivalence when:

$$R_1\sigma =_E R_2\sigma \;\Leftrightarrow\; R_1\sigma' =_E R_2\sigma' \text{ for any recipes } R_1, R_2$$

Static equivalence

Static equivalence $\sigma \sim \sigma'$ (modulo E)

Two sequences of messages $\sigma = \{w_1 \to u_1, \ldots, w_n \to u_n\}$ and $\sigma' = \{w_1 \to u'_1, \ldots, w_n \to u'_n\}$ are in static equivalence when:

$$R_1\sigma =_E R_2\sigma \;\Leftrightarrow\; R_1\sigma' =_E R_2\sigma' \text{ for any recipes } R_1, R_2$$

Example: a simple voting protocol with 3 voters a, b, and c.

$$V \to S : \langle V, \{vote\}_k \rangle$$

Static equivalence

Static equivalence $\sigma \sim \sigma'$ (modulo E)

Two sequences of messages $\sigma = \{w_1 \to u_1, \ldots, w_n \to u_n\}$ and $\sigma' = \{w_1 \to u'_1, \ldots, w_n \to u'_n\}$ are in static equivalence when:

$$R_1\sigma =_E R_2\sigma \;\Leftrightarrow\; R_1\sigma' =_E R_2\sigma' \text{ for any recipes } R_1, R_2$$

Example: a simple voting protocol with 3 voters a, b, and c.

$$V \to S : \langle V, \{vote\}_k \rangle$$

- $\sigma \;= \{w_1 \to \langle a,\ \{yes\}_k\rangle;\ w_2 \to \langle b, \{no\}_k\rangle;\ w_3 \to \langle c, \{yes\}_k\rangle\}$, and
- $\sigma' = \{w_1 \to \langle a, \{no\}_k\rangle;\ w_2 \to \langle b, \{yes\}_k\rangle;\ w_3 \to \langle c, \{yes\}_k\rangle\}$.

where a, b, c, yes, and no are public constants.

Static equivalence

Static equivalence $\sigma \sim \sigma'$ (modulo E)

Two sequences of messages $\sigma = \{w_1 \rightarrow u_1, \ldots, w_n \rightarrow u_n\}$ and $\sigma' = \{w_1 \rightarrow u'_1, \ldots, w_n \rightarrow u'_n\}$ are in static equivalence when:

$$R_1\sigma =_E R_2\sigma \;\Leftrightarrow\; R_1\sigma' =_E R_2\sigma' \text{ for any recipes } R_1, R_2$$

Example: a simple voting protocol with 3 voters a, b, and c.

$$V \rightarrow S : \langle V, \{vote\}_k \rangle$$

- $\sigma \;= \{w_1 \rightarrow \langle a,\; \{yes\}_k\rangle;\; w_2 \rightarrow \langle b, \{no\}_k\rangle;\; w_3 \rightarrow \langle c, \{yes\}_k\rangle\}$, and
- $\sigma' = \{w_1 \rightarrow \langle a, \{no\}_k\rangle;\; w_2 \rightarrow \langle b, \{yes\}_k\rangle;\; w_3 \rightarrow \langle c, \{yes\}_k\rangle\}$.

where a, b, c, yes, and no are public constants.

$\longrightarrow$ σ and σ' are not in static equivalence

$R_1 = \mathrm{proj}_2(w_1)$ and $R_2 = \mathrm{proj}_2(w_3)$

Static equivalence

Static equivalence $\sigma \sim \sigma'$ (modulo E)

Two sequences of messages $\sigma = \{w_1 \rightarrow u_1, \ldots, w_n \rightarrow u_n\}$ and $\sigma' = \{w_1 \rightarrow u'_1, \ldots, w_n \rightarrow u'_n\}$ are in static equivalence when:

$$R_1\sigma =_E R_2\sigma \;\Leftrightarrow\; R_1\sigma' =_E R_2\sigma' \text{ for any recipes } R_1, R_2$$

The static equivalence problem is decidable (even in PTIME) for many interesting equational theories useful to model cryptography primitives. [Abadi & Cortier, TCS 2006], [Cortier & Delaune, JAR 2012]

⟶ Some automatic tools are available, e.g.

- YAPA: `http://www.lsv.ens-cachan.fr/~baudet/yapa/`
- KISS: `http://www.lsv.ens-cachan.fr/~ciobaca/kiss/`
- FAST: `http://www.infsec.ethz.ch/people/brunoco`

Testing equivalence (for processes with replication)

Some decidability results [Chrétien, Cortier & D., ICALP'13 & CONCUR'14]

- \- restricted set of cryptographic primitives
- \- some syntaxic restrictions on the shape of the processes

A more pragmatic approach [Blanchet *et al.*, 2005]

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

- \+ various cryptographic primitives
- \- termination is not guaranteed; diff-equivalence (too strong)

⟶ These results are not suitable to analyse vote-privacy, or unlinkability of the BAC protocol.

Testing equivalence (for processes without replication)

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).

Testing equivalence (for processes without replication)

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], . . .

Our procedure in a nutshell

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 symbolic traces.

   ⟶ this set is still huge (exponential) !

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

   ⟶ this algorithm works quite well

Our procedure in a nutshell

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 symbolic traces.

   ⟶ this set is still huge (exponential) !

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

   ⟶ this algorithm works quite well

Some applications

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

Our procedure in a nutshell

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 symbolic traces.

   ⟶ this set is still huge (exponential) !

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

   ⟶ this algorithm works quite well

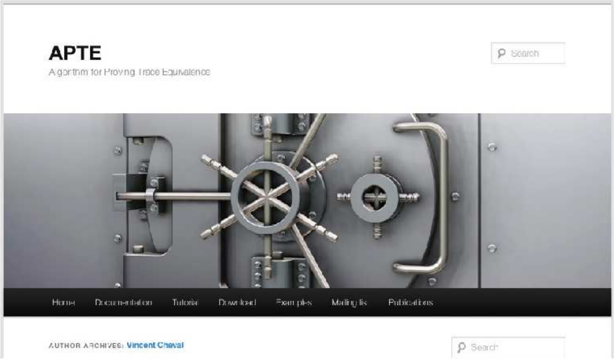
Main limitations

- e-voting protocols are still out of reach
- we can only handle very few sessions (state space explosion problem)

APTE- Algorithm for Proving Trace Equivalence

`http://projects.lsv.ens-cachan.fr/APTE`

⟶ developed by Vincent CHEVAL



APTE

Algorithm for Proving Trace Equivalence

AUTHOR ARCHIVES: Vincent Cheval

⟶ written in Ocaml, around 12 KLocs

Conclusion - What remains to do?

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

- formal definitions of some sublte security properties (receipt-freeness, coercion-resistance, . . .)
- algorithms (and tools!) for checking (automatically or not) testing equivalence for various cryptographic primitives;
- more composition results.

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

⟶ a postdoc position is available on this project.