Security protocols: from symbolic to cryptographic models

Véronique Cortier[1]

Workshop in Honour of Hubert Comon-Lundh

[1]LORIA - CNRS, INRIA Cassis project, Nancy Universities

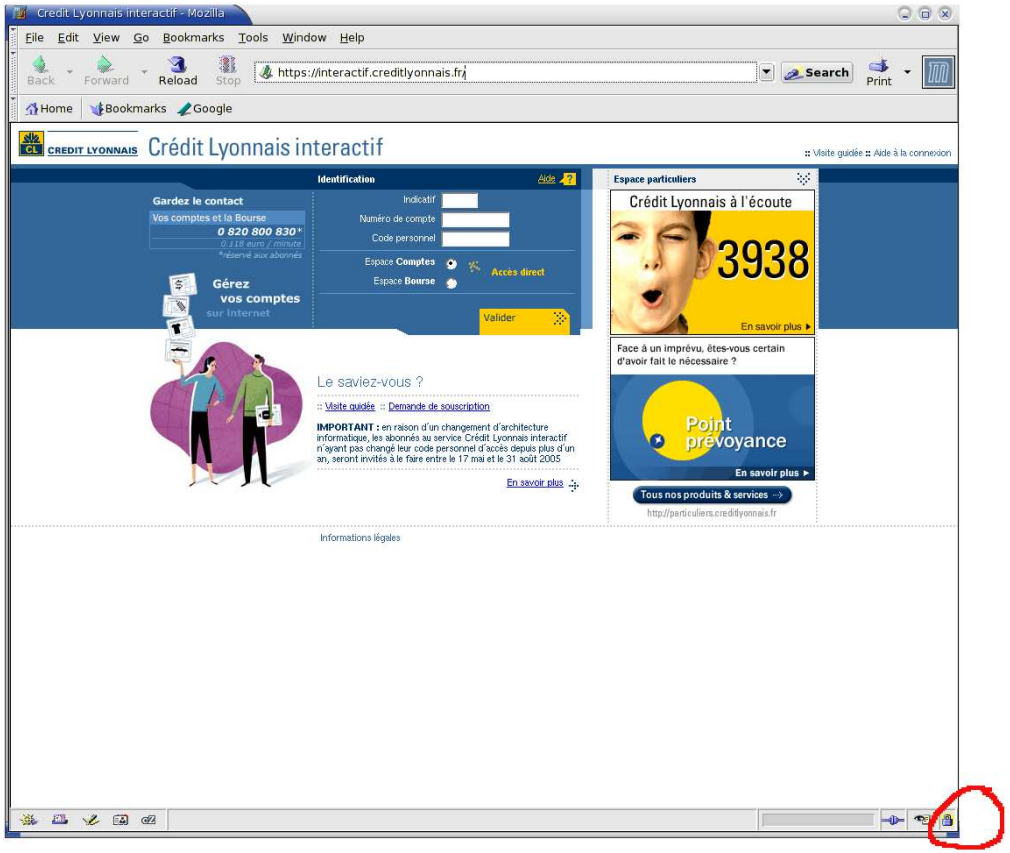
Context : cryptographic protocols

- Widely used : web (SSH, SSL, ...), pay-per-view, electronic purse, mobile phone, ...
- Should ensure : confidentiality, authenticity, integrity, anonymity, ...

Context : cryptographic protocols

- Widely used : web (SSH, SSL, ...), pay-per-view, electronic purse, mobile phone, ...
- Should ensure : confidentiality, authenticity, integrity, anonymity, ...
- Presence of an attacker
  - may read every message sent on the net,
  - may intercept and send new messages.

Security on the Internet



- confidentiality
- anonymity
- authentication

(it is really my bank ?)

Electronic voting

- The result reflects the votes
- Each vote is confidential
- Each voter can vote once and at most once
- ...

Zero-knowledge Protocol

Can you prove that you know something without revealing it ?

Example :

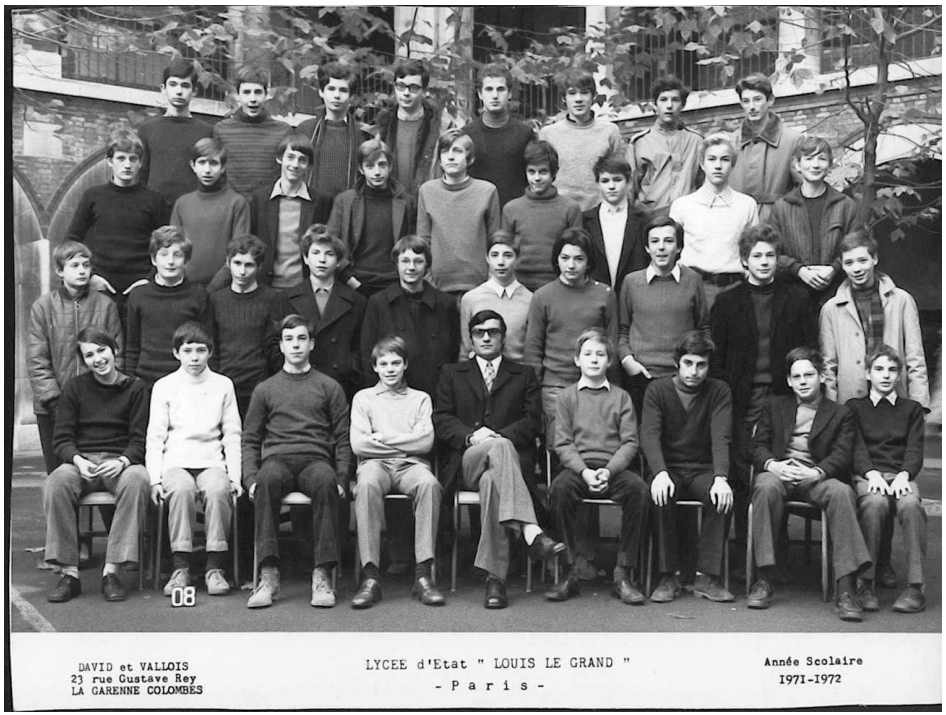



Zero-knowledge Protocol

Can you prove that you know something without revealing it ?

Example :



Can you prove that you found Hubert on a given picture ?
(This also relates to privacy on the Internet... ;-))

An attack on a Credit Card Payment Protocol

- The waiter introduces the credit card.
- The waiter enters the amount m of the transaction on the terminal.
- The terminal authenticates the card.
- The customer enters his secret code.
  If the amount m is greater than 100 euros
  (and in only 20% of the cases)
  - The terminal asks the bank for authentication of the card.
  - The bank provides authentication.

More details

4 actors : Bank, Customer, Card and Terminal.

Bank owns

- a signing key K_B^{-1}, secret,
- a verification key K_B, public,
- a secret symmetric key for each credit card K_{CB}, secret.

Card owns

- Data : last name, first name, card's number, expiration date,
- Signature's Value $VS = \{hash(\text{Data})\}_{K_B^{-1}}$,
- secret key K_{CB}.

Terminal owns the verification key K_B for bank's signatures.

Credit card payment Protocol (in short)

The terminal reads the card :

1. $Ca \rightarrow T : \mathrm{Data}, \{hash(\mathrm{Data})\}_{K_B^{-1}}$

Credit card payment Protocol (in short)

The terminal reads the card :

$$1. \quad Ca \quad \rightarrow \quad T : \mathrm{Data}, \{hash(\mathrm{Data})\}_{K_B^{-1}}$$

The terminal asks for the secret code :

$$2. \quad T \quad \rightarrow \quad Cu : secret\ code?$$
$$3. \quad Cu \quad \rightarrow \quad Ca : 1234$$
$$4. \quad Ca \quad \rightarrow \quad T : ok$$

Credit card payment Protocol (in short)

The terminal reads the card :

$$1. \quad Ca \rightarrow T : \text{Data}, \{hash(\text{Data})\}_{K_B^{-1}}$$

The terminal asks for the secret code :

$$2. \quad T \rightarrow Cu : secret\ code?$$
$$3. \quad Cu \rightarrow Ca : 1234$$
$$4. \quad Ca \rightarrow T : ok$$

The terminal calls the bank :

$$5. \quad T \rightarrow B : auth?$$
$$6. \quad B \rightarrow T : N_b$$
$$7. \quad T \rightarrow Ca : N_b$$
$$8. \quad Ca \rightarrow T : \{N_b\}_{K_{CB}}$$
$$9. \quad T \rightarrow B : \{N_b\}_{K_{CB}}$$
$$10. \quad B \rightarrow T : ok$$

Some flaws

The security was initially ensured by :

- the cards were very difficult to reproduce,
- the protocol and the keys were secret.

But

- cryptographic flaw : 320 bits keys can be broken (1988),
- logical flaw : no link between the secret code and the authentication of the card,
- fake cards can be build.

Some flaws

The security was initially ensured by :

- the cards were very difficult to reproduce,
- the protocol and the keys were secret.

But

- cryptographic flaw : 320 bits keys can be broken (1988),
- logical flaw : no link between the secret code and the authentication of the card,
- fake cards can be build.

→ "YesCard" build by Serge Humpich (1998).

How does the "YesCard" work ?

Logical flaw

1. $Ca \rightarrow T$: Data, $\{hash(\text{Data})\}_{K_B^{-1}}$
2. $T \rightarrow Ca$: *secret code?*
3. $Cu \rightarrow Ca$: 1234
4. $Ca \rightarrow T$: *ok*

How does the "YesCard" work ?

Logical flaw

1. $Ca \longrightarrow T$: Data, $\{hash(\text{Data})\}_{K_B^{-1}}$
2. $T \longrightarrow Ca$: *secret code?*
3. $Cu \longrightarrow Ca'$: 2345
4. $Ca' \longrightarrow T$: *ok*

How does the "YesCard" work ?

Logical flaw

1. $Ca \rightarrow T$: Data, $\{hash(\text{Data})\}_{K_B^{-1}}$
2. $T \rightarrow Ca$: *secret code?*
3. $Cu \rightarrow Ca'$: 2345
4. $Ca' \rightarrow T$: *ok*

Remark : there is always somebody to debit.
$\rightarrow$ creation of a fake card (Serge Humpich).

How does the "YesCard" work ?

Logical flaw

1. $Ca \rightarrow T$: Data, $\{hash(\text{Data})\}_{K_B^{-1}}$
2. $T \rightarrow Ca$: *secret code?*
3. $Cu \rightarrow Ca'$: 2345
4. $Ca' \rightarrow T$: *ok*

Remark : there is always somebody to debit.
$\rightarrow$ creation of a fake card (Serge Humpich).

1. $Ca' \rightarrow T$: XXX, $\{hash(\text{XXX})\}_{K_B^{-1}}$
2. $T \rightarrow Cu$: *secret code?*
3. $Cu \rightarrow Ca'$: 0000
4. $Ca' \rightarrow T$: *ok*

Outline of the talk

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

$$\{\text{pin} : 3443\}_{k_{\text{alice}}} \longrightarrow$$

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

$$\{\text{pin} : 3443\}_{k_{\text{alice}}} \longrightarrow$$

$$\left\{\{\text{pin} : 3443\}_{k_{\text{alice}}}\right\}_{k_{\text{bob}}} \longleftarrow$$

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

$$\{\text{pin} : 3443\}_{k_{\text{alice}}} \longrightarrow$$

$$\left\{\{\text{pin} : 3443\}_{k_{\text{alice}}}\right\}_{k_{\text{bob}}} \longleftarrow$$

$$\{\text{pin} : 3443\}_{k_{\text{bob}}} \longrightarrow$$

$$\text{Since } \left\{\{\text{pin} : 3443\}_{k_{\text{alice}}}\right\}_{k_{\text{bob}}} = \left\{\{\text{pin} : 3443\}_{k_{\text{bob}}}\right\}_{k_{\text{alice}}}$$

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

$$\{\text{pin} : 3443\}_{k_{\text{alice}}} \longrightarrow$$

$$\left\{\{\text{pin} : 3443\}_{k_{\text{alice}}}\right\}_{k_{\text{bob}}} \longleftarrow$$

$$\{\text{pin} : 3443\}_{k_{\text{bob}}} \longrightarrow$$

$\longrightarrow$ It does not work ! (Authentication problem)

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

$$\{\text{pin} : 3443\}_{k_{\text{alice}}} \longrightarrow$$

$$\left\{\{\text{pin} : 3443\}_{k_{\text{alice}}}\right\}_{k_{\text{bob}}} \longleftarrow$$

$$\{\text{pin} : 3443\}_{k_{\text{bob}}} \longrightarrow$$

→ It does not work ! (Authentication problem)

$$\{\text{pin} : 3443\}_{k_{\text{alice}}} \longrightarrow$$

$$\left\{\{\text{pin} : 3443\}_{k_{\text{alice}}}\right\}_{k_{\text{intruder}}} \longleftarrow$$

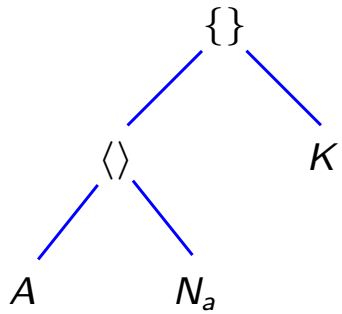
$$\{\text{pin} : 3443\}_{k_{\text{intruder}}} \longrightarrow$$

Messages

Messages are abstracted by terms.

Agents : $a, b, \ldots$ Nonces : $n_1, n_2, \ldots$ Keys : $k_1, k_2, \ldots$
Cyphertext : $\{m\}_k$ Concatenation : $\langle m_1, m_2 \rangle$

Example : The message $\{A, N_a\}_K$ is represented by :



Intruder abilities

Composition rules

$$\frac{T \vdash u \quad T \vdash v}{T \vdash \langle u\,, v\rangle} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathsf{enc}(u, v)} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathsf{enca}(u, v)}$$

Intruder abilities

Composition rules

$$\frac{T \vdash u \quad T \vdash v}{T \vdash \langle u\,, v\rangle} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathrm{enc}(u, v)} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathrm{enca}(u, v)}$$

Decomposition rules

$$\frac{}{T \vdash u}\ u \in T \qquad \frac{T \vdash \langle u\,, v\rangle}{T \vdash u} \qquad \frac{T \vdash \langle u\,, v\rangle}{T \vdash v}$$

$$\frac{T \vdash \mathrm{enc}(u, v) \quad T \vdash v}{T \vdash u} \qquad \frac{T \vdash \mathrm{enca}(u, \mathrm{pub}(v)) \quad T \vdash \mathrm{priv}(v)}{T \vdash u}$$

Intruder abilities

Composition rules

$$\frac{T \vdash u \quad T \vdash v}{T \vdash \langle u\,, v\rangle} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathrm{enc}(u, v)} \qquad \frac{T \vdash u \quad T \vdash v}{T \vdash \mathrm{enca}(u, v)}$$

Decomposition rules

$$\frac{}{T \vdash u}\ u \in T \qquad \frac{T \vdash \langle u\,, v\rangle}{T \vdash u} \qquad \frac{T \vdash \langle u\,, v\rangle}{T \vdash v}$$

$$\frac{T \vdash \mathrm{enc}(u, v) \quad T \vdash v}{T \vdash u} \qquad \frac{T \vdash \mathrm{enca}(u, \mathrm{pub}(v)) \quad T \vdash \mathrm{priv}(v)}{T \vdash u}$$

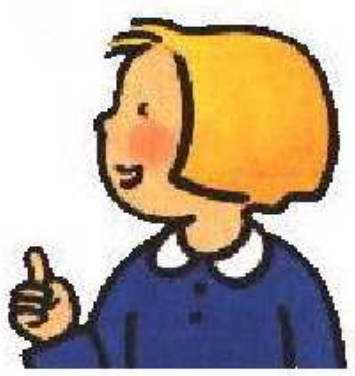
Deducibility relation

A term u is deducible from a set of terms T, denoted by $T \vdash u$, if there exists a prooftree witnessing this fact.

A simple protocol

$\langle \text{Bob}, k \rangle$ →

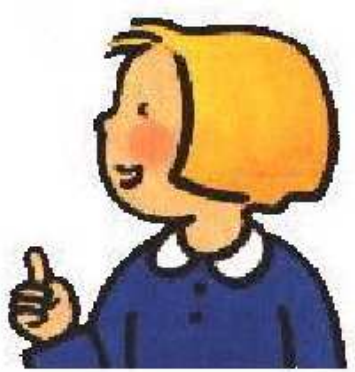
← $\langle \text{Alice}, \text{enc}(s, k) \rangle$

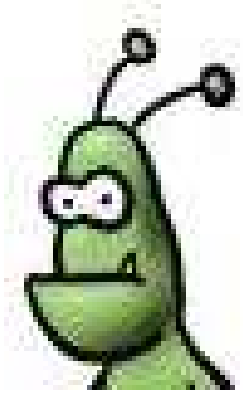


A simple protocol

$\langle Bob, k \rangle$ →

← $\langle Alice, enc(s, k) \rangle$





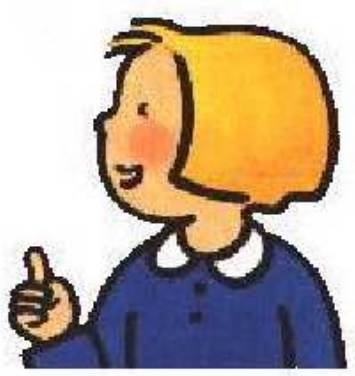
Question ?

Can the attacker learn the secret s ?

A simple protocol

$\langle Bob, k \rangle \longrightarrow$

$\longleftarrow \langle Alice, enc(s, k) \rangle$



Answer : Of course, Yes !

$$\frac{\langle Alice, enc(s, k) \rangle}{enc(s, k)} \qquad \frac{\langle Bob, k \rangle}{k}$$

$$\frac{\qquad\qquad\qquad\qquad\qquad\qquad}{s}$$

Decision of the intruder problem

Given A set of messages S and a message m

Question Can the intruder learn m from S that is $S \vdash m$?

This problem is decidable in polynomial time

Decision of the intruder problem

Given A set of messages S and a message m

Question Can the intruder learn m from S that is $S \vdash m$?

This problem is decidable in polynomial time

Lemma (Locality)

If there is a proof of $S \vdash m$ then there is a proof that only uses the subterms of S and m.

Protocol description

Protocol :

$$\begin{array}{lcl} A \to B & : & \{\mathsf{pin}\}_{k_a} \\ B \to A & : & \{\{\mathsf{pin}\}_{k_a}\}_{k_b} \\ A \to B & : & \{\mathsf{pin}\}_{k_b} \end{array}$$

A protocol is a finite set of roles :

- role $\Pi(1)$ corresponding to the 1^{st} participant played by a talking to b :

$$\begin{array}{rcl} \mathsf{init} & \xrightarrow{k_a} & \mathsf{enc}(\mathsf{pin}, k_a) \\ \mathsf{enc}(x, k_a) & \to & x. \end{array}$$

Protocol description

Protocol :

$$\begin{array}{lcl} A \to B & : & \{\mathsf{pin}\}_{k_a} \\ B \to A & : & \{\{\mathsf{pin}\}_{k_a}\}_{k_b} \\ A \to B & : & \{\mathsf{pin}\}_{k_b} \end{array}$$

A protocol is a finite set of roles :

- role $\Pi(1)$ corresponding to the 1^{st} participant played by a talking to b :

$$\begin{array}{rcl} \mathsf{init} & \xrightarrow{k_a} & \mathsf{enc}(\mathsf{pin}, k_a) \\ \mathsf{enc}(x, k_a) & \to & x. \end{array}$$

- role $\Pi(2)$ corresponding to the 2^{nd} participant played by b with a :

$$\begin{array}{rcl} x & \xrightarrow{k_b} & \mathsf{enc}(x, k_b) \\ \mathsf{enc}(y, k_b) & \to & \mathsf{stop}. \end{array}$$

Secrecy via constraint solving

Constraint systems are used to specify secrecy preservation under a particular, finite scenario.

Scenario

$$\begin{array}{l} \mathrm{rcv}(u_1) \xrightarrow{N_1} \mathrm{snd}(v_1) \\ \mathrm{rcv}(u_2) \xrightarrow{N_2} \mathrm{snd}(v_2) \\ \quad \ldots \\ \mathrm{rcv}(u_n) \xrightarrow{N_n} \mathrm{snd}(v_n) \end{array}$$

Constraint System

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

Remark : Constraint Systems may be used more generally for trace-based properties, e.g. authentication.

Secrecy via constraint solving

Constraint systems are used to specify secrecy preservation under a particular, finite scenario.

Scenario

$$\begin{array}{l} \mathrm{rcv}(u_1) \xrightarrow{N_1} \mathrm{snd}(v_1) \\ \mathrm{rcv}(u_2) \xrightarrow{N_2} \mathrm{snd}(v_2) \\ \quad \ldots \\ \mathrm{rcv}(u_n) \xrightarrow{N_n} \mathrm{snd}(v_n) \end{array}$$

Constraint System

$$\mathcal{C} = \left\{ \begin{array}{l} T_0 \Vdash u_1 \\ T_0, v_1 \Vdash u_2 \\ \quad \ldots \\ T_0, v_1, .., v_n \Vdash s \end{array} \right.$$

Solution of a constraint system

A substitution σ such that

for every $T \Vdash u \in \mathcal{C}$, $u\sigma$ is deducible from $T\sigma$, that is $u\sigma \vdash T\sigma$.

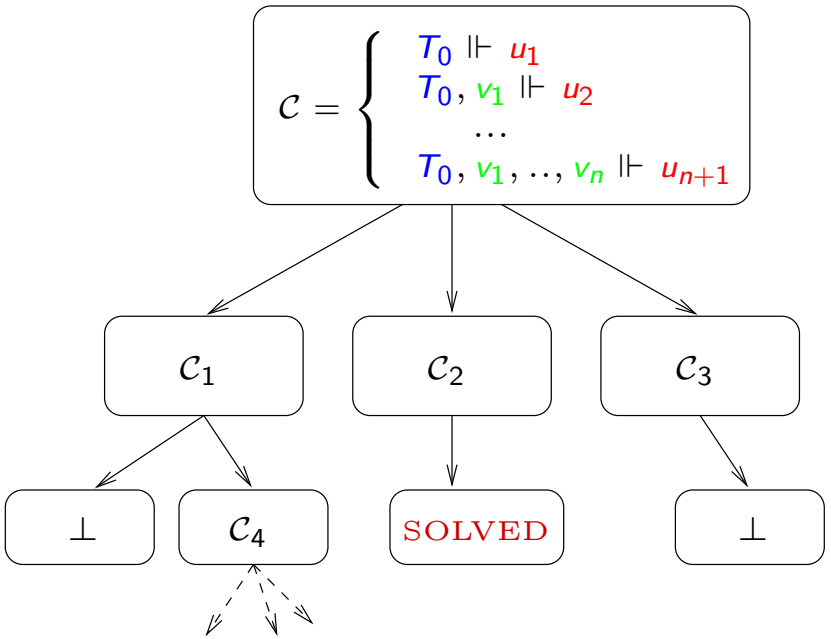
How to solve constraint system ?

$$\text{Given } \mathcal{C} = \begin{cases} T_0 \Vdash u_1 \\ T_0, v_1 \Vdash u_2 \\ \quad \dots \\ T_0, v_1, .., v_n \Vdash u_{n+1} \end{cases}$$

Question Is there a solution σ of $\mathcal{C}$?

Decision procedure [Comon-Lundh]

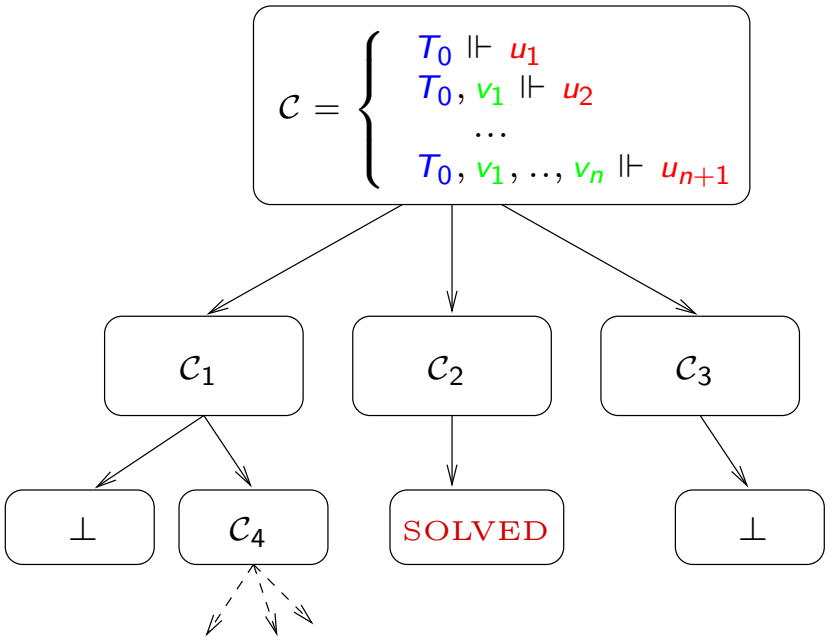
Hubert's approach : Transforming the constraints into simpler (solved) ones.



$\mathcal{C}$ has a solution iff $\mathcal{C} \rightsquigarrow \mathcal{C}'$ with $\mathcal{C}'$ in solved form.

Decision procedure [Comon-Lundh]

Hubert's approach : Transforming the constraints into simpler (solved) ones.



$\mathcal{C}$ has a solution iff $\mathcal{C} \rightsquigarrow \mathcal{C}'$ with $\mathcal{C}'$ in solved form.

Advertisement : Lecture of Ralf Treinen tomorrow.

Advantages of the approach

- Can be adapted to other cryptographic primitives, possibly adding equational theories
- Provides a complete (finite and symbolic) characterization of all possible behaviors
  $\longrightarrow$ enables to verify many security properties (e.g. authentication, secrecy, key cycles, timestamps, ...)

Limits : bounded number of sessions

What formal methods allow to do ?

- In general, secrecy preservation is undecidable.

What formal methods allow to do ?

- In general, secrecy preservation is undecidable.
- For a bounded number of sessions, secrecy is co-NP-complete [RusinowitchTuruani CSFW01]
  $\longrightarrow$ numerous tools for detecting attacks (Casper, Avispa platform...)

What formal methods allow to do ?

- In general, secrecy preservation is undecidable.
- For a bounded number of sessions, secrecy is co-NP-complete [RusinowitchTuruani CSFW01]
  $\longrightarrow$ numerous tools for detecting attacks (Casper, Avispa platform...)
- For an unbounded number of sessions
  - for one-copy protocols, secrecy is DEXPTIME-complete [Comon-Lundh&Cortier RTA03] [SeildVerma LPAR04]
  - for message-length bounded protocols, secrecy is DEXPTIME-complete [Durgin et al FMSP99] [Chevalier et al CSL03]

  $\longrightarrow$ some tools for proving security (ProVerif, EVA Platform)

Tools

Many tools for a bounded number of sessions (search for attacks) :
Casper, Avispa platform, ...

Some tools for an unbounded number of sessions (security proof) :
ProVerif, EVA platform

- new attacks have been discovered (e.g. the man-in-the-middle attack on the Needham-Schroeder protocol)
- hundreds protocols analyzed in few minutes or few seconds for most of them
- real-world applications (IETF, ...)

Outline of the talk

Motivation

Back to our running example :

$$\begin{array}{lcl} A \to B & : & \{\mathsf{pin}\}_{k_a} \\ B \to A & : & \{\{\mathsf{pin}\}_{k_a}\}_{k_b} \\ A \to B & : & \{\mathsf{pin}\}_{k_b} \end{array}$$

We need the equation for the commutativity of encryption

$$\{\{z\}_x\}_y = \{\{z\}_y\}_x$$

Some other examples

Encryption-Decryption theory

$$\mathsf{dec}(\mathsf{enc}(x,y),y) = x \quad \pi_1(\langle x,y\rangle) = x \quad \pi_2(\langle x,y\rangle) = y$$

EXclusive Or

$$\begin{array}{rcrcrcl} x \oplus (y \oplus z) & = & z & & x \oplus y & = & y \oplus x \\ x \oplus x & = & 0 & & x \oplus 0 & = & x \end{array}$$

Diffie-Hellmann

$$\exp(\exp(z,x),y) = \exp(\exp(z,y),x)$$

E-voting protocols

First phase :

$$V \to A : \quad \mathsf{sign}(\mathit{blind}(\mathsf{vote}, r), V)$$
$$A \to V : \quad \mathsf{sign}(\mathit{blind}(\mathsf{vote}, r), A)$$

Voting phase :

$$V \to C : \quad \mathsf{sign}(\mathsf{vote}, A)$$

...

Equational theory for blind signatures

[Kremer Ryan 05]

$$\begin{aligned} \mathsf{checksign}(\mathsf{sign}(x,y),\mathsf{pk}(y)) &= x \\ \mathsf{unblind}(\mathsf{blind}(x,y),y) &= x \\ \mathsf{unblind}(\mathsf{sign}(\mathsf{blind}(x,y),z),y) &= \mathsf{sign}(x,z) \end{aligned}$$

Deduction

$$\frac{}{T \vdash_E M} M \in T \qquad \frac{T \vdash_E M_1 \quad \cdots \quad T \vdash_E M_k}{T \vdash_E f(M_1, \ldots, M_k)} f \in \Sigma$$

$$\frac{T \vdash M}{T \vdash M'} M =_E M'$$

Deduction

$$\frac{}{T \vdash_E M} M \in T \qquad \frac{T \vdash_E M_1 \quad \cdots \quad T \vdash_E M_k}{T \vdash_E f(M_1, \ldots, M_k)} f \in \Sigma$$

$$\frac{T \vdash M}{T \vdash M'} M =_E M'$$

Example : $E := \mathrm{dec}(\mathrm{enc}(x, y), y) = x$ and $T = \{\mathrm{enc}(\mathit{secret}, k),\ k\}$.

$$\dfrac{\dfrac{\dfrac{}{T \vdash \mathrm{enc}(\mathit{secret}, k)} \qquad \dfrac{}{T \vdash k}}{T \vdash \mathrm{dec}(\mathrm{enc}(\mathit{secret}, k), k)}}{T \vdash \mathit{secret}} \quad f \in \Sigma \quad \mathrm{dec}(\mathrm{enc}(x, y), y) = x$$

Rewriting systems

For analyzing equational theories, we (try to) associate to E a finite convergent rewriting system $\mathcal{R}$ such that :

$$u =_E v \quad \text{iff} \quad u\downarrow = v\downarrow$$

Definition (Characterization of the deduction relation)

Let $t_1, \ldots t_n$ and u be terms in normal form.

$$\{t_1, \ldots t_n\} \vdash u \quad \text{iff} \quad \exists C \text{ s.t. } C[t_1, \ldots, t_n] \rightarrow^* u$$

(Also called Cap Intruder problem [Narendran et al])

Active case

Hubert's approach

1. Reducing the equational theory E to AC :

$$\begin{cases} T_0 \Vdash_E u_1 \\ T_0, v_1 \Vdash_E u_2 \\ \quad \dots \\ T_0, v_1, .., v_n \Vdash_E u_{n+1} \end{cases} \rightsquigarrow \begin{cases} T'_0 \Vdash_{AC} u'_1 \\ T'_0, v'_1 \Vdash_{AC} u'_2 \\ \quad \dots \\ T'_0, v'_1, .., v'_n \Vdash_{AC} u'_{n+1} \end{cases}$$

guessing in advance the equalities (finite variant property) [Comon-Lundh&Delaune]

2. Solving the AC case (not so easy...) [Bursuc&Comon-Lundh&Delaune]

Outline of the talk

Specificity of cryptographic models

- Messages are bitstrings
- Real encryption algorithm
- Real signature algorithm
- General and powerful adversary

$\longrightarrow$ very little abstract model

Encryption nowadays

$\longrightarrow$ Based on algorithmically hard problems.

RSA Function $n = pq$, p et q primes.
e : public exponent

- $x \mapsto x^e \mod n$ easy (cubic)
- $y = x^e \mapsto x \mod n$ difficult
  $x = y^d$ où $d = e^{-1} \mod \phi(n)$

Encryption nowadays

$\longrightarrow$ Based on algorithmically hard problems.

RSA Function $n = pq$, p et q primes.
e : public exponent

- $x \mapsto x^e \mod n$ easy (cubic)
- $y = x^e \mapsto x \mod n$ difficult
  $x = y^d$ où $d = e^{-1} \mod \phi(n)$

Diffie-Hellman Problem

- Given $A = g^a$ and $B = g^b$,
- Compute $\mathrm{DH}(A, B) = g^{ab}$

Encryption nowadays

$\longrightarrow$ Based on algorithmically hard problems.

RSA Function $n = pq$, p et q primes.
e : public exponent

- $x \mapsto x^e \mod n$ easy (cubic)
- $y = x^e \mapsto x \mod n$ difficult
  $x = y^d$ où $d = e^{-1} \mod \phi(n)$

Diffie-Hellman Problem

- Given $A = g^a$ and $B = g^b$,
- Compute $\mathrm{DH}(A, B) = g^{ab}$

$\longrightarrow$ Based on hardness of integer factorization.

Setting for cryptographic protocols

Protocol :

- Message exchange program
- using cryptographic primitives

Adversary $\mathcal{A}$: any probabilistic polynomial Turing machine, *i.e.* any probabilistic polynomial program.

- polynomial : captures what is feasible
- probabilistic : the adversary may try to guess some information

Definition of secrecy preservation

→ Several notions of secrecy :

One-Wayness : The probability for an adversary $\mathcal{A}$ to compute the secret s against a protocol $\mathcal{P}$ is negligible (smaller than any inverse of polynomial).

$$\forall p \text{ polynomial } \exists \eta_0 \, \forall \eta \geq \eta_0 \quad \Pr^{\eta}_{m,r}[\mathcal{A}(\mathcal{P}_K) = s] \leq \frac{1}{p(\eta)}$$

η : security parameter = key length

Not strong enough !

- The adversary may be able to compute half of the secret message.
- There is no guarantee in case that some partial information on the secret is known.

Not strong enough !

- The adversary may be able to compute half of the secret message.
- There is no guarantee in case that some partial information on the secret is known.

→ Introduction of a notion of indistinguishability.

Indistinguishability

The secrecy of s is defined through the following game :

- Two values n_0 and n_1 are randomly generated instead of s ;
- The adversary interacts with the protocol where s is replaced by n_b, $b \in \{0, 1\}$;
- We give the pair (n_0, n_1) to the adversary ;
- The adversary gives b',

The data s is secret if $\Pr[b = b'] - \frac{1}{2}$ is a negligible function.

Formal and Cryptographic approaches

	Formal approach	Cryptographic approach
Messages	terms	bitstrings
Encryption	idealized	algorithm
Adversary	idealized	any polynomial algorithm
Secrecy property	reachability-based property	indistinguishability
Guarantees	unclear	strong
Protocol	complex, several sessions	simple, one session

Formal and Cryptographic approaches

	Formal approach	Cryptographic approach
Messages	terms	bitstrings
Encryption	idealized	algorithm
Adversary	idealized	any polynomial algorithm
Secrecy property	reachability-based property	indistinguishability
Guarantees	unclear	strong
Protocol	complex, several sessions	simple, one session
Proof	automatic	by hand, tedious and error-prone

Link between the two approaches ?

A first result

Theorem (Micciancio&Warinschi)

Any concrete execution trace produced by a real adversary is actually captured by a symbolic trace of an symbolic adversary (except with negligible probability)

- For protocols with only public key encryption, signatures and nonces
- Provided the public key encryption and the signature algorithms verify strong existing cryptographic properties (IND-CCA2, existentially unforgeable),

Limits of existing results

All existing results only allow to transfer trace properties (+ some particular cases of secrecy)

Fact 1 : Computational security properties are often stated as indistinguishability games rather than trace properties.
Example : secrecy, ideal functionalities, ...

Limits of existing results

All existing results only allow to transfer trace properties (+ some particular cases of secrecy)

Fact 1 : Computational security properties are often stated as indistinguishability games rather than trace properties.
Example : secrecy, ideal functionalities, ...

Fact 2 : Some security properties cannot be expressed as trace properties.
Example : Privacy properties of e-voting protocols

$$P(A, a) \| P(B, b) \sim_o P(A, b) \| P(B, a)$$

Hubert's idea

Definition (Computational indistinguishability)

$P \approx Q$ if for any adversary $\mathcal{A}$ (that is any PPT Turing machine)
$|\Pr\{r, r'(P(r)\|\mathcal{A}(r')) = 1\}| - |\Pr\{r, r'(Q(r)\|\mathcal{A}(r')) = 1\}|$
is negligible.

Intuitively, an attacker cannot tell the difference between P and Q.

Hubert's idea

Definition (Computational indistinguishability)

$P \approx Q$ if for any adversary $\mathcal{A}$ (that is any PPT Turing machine)
$|\Pr\{r, r'(P(r)\|\mathcal{A}(r')) = 1\}| - |\Pr\{r, r'(Q(r)\|\mathcal{A}(r')) = 1\}|$
is negligible.

Intuitively, an attacker cannot tell the difference between P and Q.

There exists a similar symbolic definition !

Definition (observational equivalence)

$P \sim_o Q$ if for any process O, we have $P\|O \sim Q\|O$.

Intuitively, an observer cannot tell the difference between P and Q.

Soundness of observational equivalence

Observational equivalence is a sound abstraction of computational indistinguishability.

$$P \sim_o Q \Rightarrow [\![P]\!] \approx [\![Q]\!]$$

- For simple processes
  (A fragment of applied pi-calculus that captures most security protocols)
- For symmetric encryption implemented using IND-CC2 schemes

Soundness of observational equivalence

Observational equivalence is a sound abstraction of computational indistinguishability.

$$P \sim_o Q \Rightarrow [\![P]\!] \approx [\![Q]\!]$$

- For simple processes
  (A fragment of applied pi-calculus that captures most security protocols)
- For symmetric encryption implemented using IND-CC2 schemes

Applications :

- symbolic proof of privacy-like properties
- symbolic proof of anonymity
- symbolic proof of simulatability
- symbolic proof of secrecy (to some extend)

Conclusion

Formal methods, including of course rewriting techniques, form a very powerful approach for analyzing security protocols

- Many automatic tools (ProVerif, Avispa, ...)
- Cryptographic guarantees

Conclusion

Formal methods, including of course rewriting techniques, form a very powerful approach for analyzing security protocols

- Many automatic tools (ProVerif, Avispa, ...)
- Cryptographic guarantees

Some current directions of research :

- Considering more equational theories (e.g. theories for e-voting protocols)
- Combining formal and cryptographic models
- Adding more complex structures for data (list, XML, ...)
- Considering more complex execution systems (loops, ...)