Verification techniques for cryptographic protocols

Véronique Cortier[1]

RTA'08

[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.

Example : 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 :

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

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 :

$$\begin{array}{llll} 2. & T & \rightarrow & Cu : secret\ code? \\ 3. & Cu & \rightarrow & Ca : 1234 \\ 4. & Ca & \rightarrow & T : ok \end{array}$$

Credit card payment Protocol (in short)

The terminal reads the card :

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

The terminal asks for the secret code :

$$\begin{array}{llll} 2. & T & \rightarrow & Cu : secret\ code? \\ 3. & Cu & \rightarrow & Ca : 1234 \\ 4. & Ca & \rightarrow & T : ok \end{array}$$

The terminal calls the bank :

$$\begin{array}{llll} 5. & T & \rightarrow & B : auth? \\ 6. & B & \rightarrow & T : N_b \\ 7. & T & \rightarrow & Ca : N_b \\ 8. & Ca & \rightarrow & T : \{N_b\}_{K_{CB}} \\ 9. & T & \rightarrow & B : \{N_b\}_{K_{CB}} \\ 10. & B & \rightarrow & T : ok \end{array}$$

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

$$
\begin{array}{llcll}
1. & Ca & \rightarrow & T & : \mathrm{Data}, \{hash(\mathrm{Data})\}_{K_B^{-1}} \\
2. & T & \rightarrow & Ca & : secret\ code? \\
3. & Cu & \rightarrow & Ca & : 1234 \\
4. & Ca & \rightarrow & T & : ok
\end{array}
$$

How does the “YesCard” work ?

Logical flaw

$$
\begin{array}{llll}
1. & Ca & \rightarrow T & : \mathrm{Data}, \{hash(\mathrm{Data})\}_{K_B^{-1}} \\
2. & T & \rightarrow Ca & : secret\ code? \\
3. & Cu & \rightarrow Ca' & : 2345 \\
4. & Ca' & \rightarrow T & : ok
\end{array}
$$

How does the "YesCard" work ?

Logical flaw

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

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

How does the "YesCard" work ?

Logical flaw

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

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

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

Outline of the talk

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

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

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

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

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

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

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

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

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

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

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

$$\xrightarrow{\{\mathrm{pin} : 3443\}_{k_{alice}}}$$

$$\xleftarrow{\left\{\{\mathrm{pin} : 3443\}_{k_{alice}}\right\}_{k_{bob}}}$$

$$\xrightarrow{\{\mathrm{pin} : 3443\}_{k_{bob}}}$$

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

Motivation : Cryptography does not suffice to ensure security !

Example : Commutative encryption (RSA)

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

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

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

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

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

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

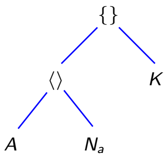
$$\xrightarrow{\{\text{pin} : 3443\}_{k_{\text{intruder}}}}$$

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

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

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

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

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

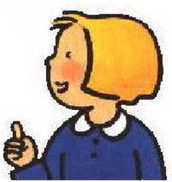
Deducibility relation

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

$$\xrightarrow{\langle \mathsf{Bob}, \mathsf{k} \rangle}$$

$$\xleftarrow{\langle \mathsf{Alice}, \mathsf{enc}(\mathsf{s}, \mathsf{k}) \rangle}$$



A simple protocol

$\xrightarrow{\langle \mathsf{Bob}, \mathsf{k} \rangle}$

$\xleftarrow{\langle \mathsf{Alice}, \mathsf{enc}(\mathsf{s}, \mathsf{k}) \rangle}$

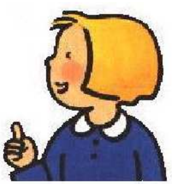
Question ?

Can the attacker learn the secret s ?

A simple protocol

$$\xrightarrow{\langle \mathsf{Bob}, \mathsf{k}\rangle}$$

$$\xleftarrow{\langle \mathsf{Alice}, \mathsf{enc}(\mathsf{s}, \mathsf{k})\rangle}$$



Answer : Of course, Yes !

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

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 :

$$\begin{array}{rcl} A \rightarrow B & : & \{\text{pin}\}_{k_a} \\ B \rightarrow A & : & \{\{\text{pin}\}_{k_a}\}_{k_b} \\ A \rightarrow B & : & \{\text{pin}\}_{k_b} \end{array}$$

A protocol is a finite set of roles :

- role $\Pi(1)$ corresponding to the 1[st] participant played by a talking to b :

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

Protocol description

Protocol :

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

A protocol is a finite set of roles :

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

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

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

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

Secrecy via constraint solving

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

Scenario

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

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

$$\begin{array}{c} \mathsf{rcv}(u_1) \xrightarrow{N_1} \mathsf{snd}(v_1) \\ \mathsf{rcv}(u_2) \xrightarrow{N_2} \mathsf{snd}(v_2) \\ \dots \\ \mathsf{rcv}(u_n) \xrightarrow{N_n} \mathsf{snd}(v_n) \end{array}$$

Constraint System

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

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 ?

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

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

How to solve constraint system ?

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 $\mathcal{C}$?

Advertisement :
Lecture of Hubert Comon-Lundh at ISR 2008 next week

An easy case : "solved constraint systems"

Given $\mathcal{C} = \begin{cases} T_0 \Vdash x_1 \\ T_0, v_1 \Vdash x_2 \\ \quad \ldots \\ T_0, v_1, .., v_n \Vdash x_{n+1} \end{cases}$

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

An easy case : “solved constraint systems”

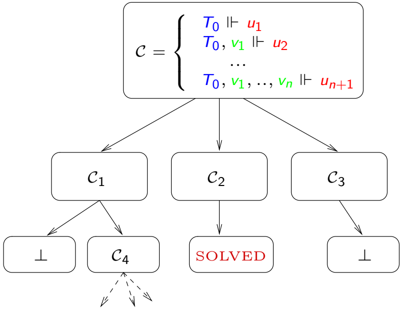
Given $\mathcal{C} = \begin{cases} T_0 \Vdash x_1 \\ T_0, v_1 \Vdash x_2 \\ \quad \dots \\ T_0, v_1, .., v_n \Vdash x_{n+1} \end{cases}$

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

Of course yes !
Consider e.g. $\sigma(x_1) = \cdots = \sigma(x_{n+1}) = t \in T_0$.

Decision procedure [Millen / Comon-Lundh]

Goal : Transformation of the constraints in order to obtain a solved constraint system.



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

Intruder step

The intruder can built messages

$$R_5 : \quad \mathcal{C} \wedge T \Vdash f(u, v) \quad \rightsquigarrow \quad \mathcal{C} \wedge T \Vdash u \wedge T \Vdash v$$
$$\text{for } f \in \{\langle\rangle, \mathrm{enc}, \mathrm{enca}\}$$

Intruder step

The intruder can built messages

$$R_5: \quad \mathcal{C} \wedge T \Vdash f(u,v) \quad \rightsquigarrow \quad \mathcal{C} \wedge T \Vdash u \wedge T \Vdash v$$
$$\text{for } f \in \{\langle\rangle, \mathrm{enc}, \mathrm{enca}\}$$

Example :

$$a, k \Vdash \mathrm{enc}(\langle x, y\rangle, k) \quad \rightsquigarrow \quad \begin{array}{l} a, k \Vdash k \\ a, k \Vdash x \\ a, k \Vdash y \end{array}$$

Eliminating redundancies

$$k \Vdash x$$
$$\mathsf{enc}(s, x) \Vdash s$$

The constraint $\mathsf{enc}(s, x) \Vdash s$ will be satisfied as soon as $k \Vdash x$ is satisfied.

Eliminating redundancies

$$k \Vdash x$$
$$\mathrm{enc}(s, x) \Vdash s$$

The constraint $\mathrm{enc}(s, x) \Vdash s$ will be satisfied as soon as $k \Vdash x$ is satisfied.

$$R_1 : \mathcal{C} \wedge T \Vdash u \rightsquigarrow \mathcal{C} \quad \text{if } T \cup \{x \mid T' \Vdash x \in \mathcal{C}, T' \subsetneq T\} \vdash u$$

Unsolvable constraints

$$R_4 : \mathcal{C} \wedge T \Vdash u \rightsquigarrow \bot \qquad \text{if } \mathrm{var}(T, u) = \emptyset \text{ and } T \not\vdash u$$

Example :

$$\begin{array}{lcl} \ldots & & \\ a, \mathrm{enc}(s, k) \Vdash s & \rightsquigarrow & \bot \\ \ldots & & \end{array}$$

Guessing equalities

1. Example : $k, \mathrm{enc}(\mathrm{enc}(x, k'), k) \Vdash \mathrm{enc}(a, k')$

$$R_2 : \mathcal{C} \ \wedge \ T \Vdash u \rightsquigarrow_\sigma \mathcal{C}\sigma \ \wedge \ T\sigma \Vdash u\sigma \qquad u' \in st(T)$$
$$\text{if } \sigma = \mathrm{mgu}(u, u'), \ u, u' \notin \mathcal{X}, \ u \neq u'$$

Guessing equalities

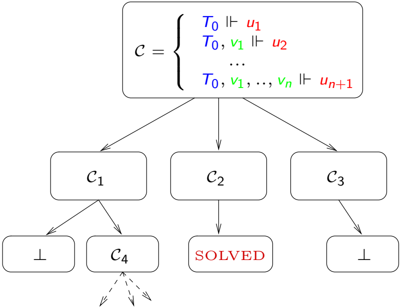
1. Example : $k, \mathrm{enc}(\mathrm{enc}(x, k'), k) \Vdash \mathrm{enc}(a, k')$

$$R_2 : \mathcal{C} \wedge T \Vdash u \rightsquigarrow_\sigma \mathcal{C}\sigma \wedge T\sigma \Vdash u\sigma \qquad u' \in st(T)$$
$$\text{if } \sigma = \mathrm{mgu}(u, u'),\ u, u' \notin \mathcal{X},\ u \neq u'$$

2. Example : $\mathrm{enc}(s, \langle a, x \rangle), \mathrm{enc}(\langle y, b \rangle, k), k \Vdash s$

$$R_3 : \mathcal{C} \wedge T \Vdash v \rightsquigarrow_\sigma \mathcal{C}\sigma \wedge T\sigma \Vdash v\sigma \qquad u, u' \in st(T)$$
$$\text{if } \sigma = \mathrm{mgu}(u, u'),\ u, u' \notin \mathcal{X},\ u \neq u'$$

NP-procedure for solving constraint systems



Theorem

- *$\mathcal{C}$ has a solution iff $\mathcal{C} \rightsquigarrow \mathcal{C}'$ with $\mathcal{C}'$ in solved form.*
- *$\rightsquigarrow$ is terminating in polynomial time.*

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]
  → 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]
  → numerous tools for detecting attacks (Casper, Avispa platform...)
- For an unbounded number of sessions
  - for one-copy protocols, secrecy is DEXPTIME-complete [CortierComon RTA03] [SeildVerma LPAR04]
  - for message-length bounded protocols, secrecy is DEXPTIME-complete [Durgin et al FMSP99] [Chevalier et al CSL03]

  → some tools for proving security (ProVerif, EVA Platform)

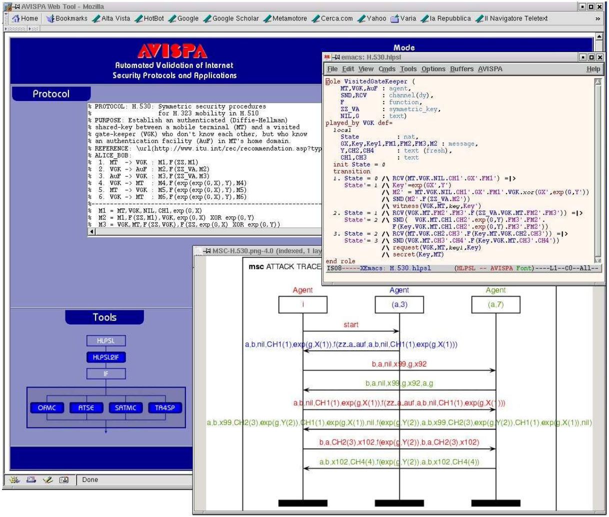
Tools

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, ...)

Example of tool : Avispa Platform



Collaborators

- Cassis project, Loria
- DIST, Italy
- ETHZ, Swiss
- Siemens, Germany

www.avispa-project.org

Outline of the talk

Motivation

Back to our running example :

$$\begin{array}{rcl} A \rightarrow B & : & \{\mathrm{pin}\}_{k_a} \\ B \rightarrow A & : & \{\{\mathrm{pin}\}_{k_a}\}_{k_b} \\ A \rightarrow B & : & \{\mathrm{pin}\}_{k_b} \end{array}$$

We need the equation for the commutativity of encryption

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

Some other examples

Encryption-Decryption theory

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

EXclusive Or

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

Diffie-Hellmann

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

E-voting protocols

First phase :

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

Voting phase :

$$V \rightarrow C : \quad \mathrm{sign}(\mathrm{vote}, A)$$

...

Equational theory for blind signatures

[Kremer Ryan 05]

$$\begin{array}{rcl}
\mathrm{checksign}(\mathrm{sign}(x,y),\mathrm{pk}(y)) & = & x \\
\mathrm{unblind}(\mathrm{blind}(x,y),y) & = & x \\
\mathrm{unblind}(\mathrm{sign}(\mathrm{blind}(x,y),z),y) & = & \mathrm{sign}(x,z)
\end{array}$$

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 $T = \{\mathrm{enc}(secret, k),\ k\}$.

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

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])

Some results with equational theories

	Security problem	
	Bounded number of sessions	**Unbounded number of sessions**
Commutative encryption	*co-NP-complete* [CKRT04]	**Ping-pong protocols :** *co-NP-complete* [Turuani04]
Exclusive Or	*Decidable* [CS03,CKRT03]	**One copy - No nonces :** *Decidable* [CLC03] **Two-way automata - No nonces :** *Decidable* [Verma03]
Abelian Groups	*Decidable* [Shmatikov04]	**Two-way automata - No nonces :** *Decidable* [Verma03]
Prefix encryption	*co-NP-complete* [CKRT03]	
Abelian Groups and Modular Exponentiation	**General case :** *Decidable* [Shmatikov04] **Restricted protocols :** *co-NP-complete* [CKRT03]	**AC properties of the Modular Exponentiation No nonces :** *Semi-Decision Procedure* [GLRV04]

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

→ 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

$\rightarrow$ 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

→ 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 $\mathrm{DH}(A, B) = g^{ab}$

→ 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).

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

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

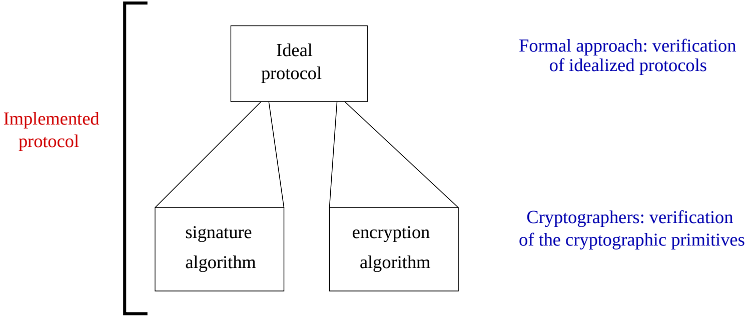
Formal and Cryptographic approaches

	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 ?

Composition of the two approaches

Automatic cryptographically sound proofs



→ Currently implemented in the Avispa platform.

Example : correspondence of secrecy properties

Theorem

Symbolic secrecy implies cryptographic indistinguishability.

- 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),

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, ...)
- ...