Security via constraint solving

Stéphanie Delaune

October 30, 2006

Cryptographic protocols

Cryptographic protocols

- small programs designed to secure communication
- use cryptographic primitives (*e.g.* encryption, hash function, . . .)

Goals

- Secrecy: May an intruder learn some secret message between two honest participants ?
- Authentication: Is the agent Alice really talking to Bob ?

Goals

- Secrecy: May an intruder learn some secret message between two honest participants ?
- Authentication: Is the agent Alice really talking to Bob ?
- Fairness: Alice and Bob want to sign a contract. Alice initiates the protocol. May Bob obtain some advantage ?
- Privacy: Alice participate to an election. May a participant learn something about the vote of Alice ?
- Receipt-Freeness: Alice participate to an election. Does Alice gain any information (a receipt) which can be used to prove to a coercer that she voted in a certain way ?
- ...

Needham-Schroeder's Protocol (1978)

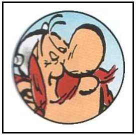
$$\begin{array}{rcrl} A & \rightarrow & B: & \{A, N_a\}_{\text{pub}(B)} \\ B & \rightarrow & A: & \{N_a, N_b\}_{\text{pub}(A)} \\ A & \rightarrow & B: & \{N_b\}_{\text{pub}(B)} \end{array}$$

Needham-Schroeder's Protocol (1978)

$$A \to B : \{A, N_a\}_{\mathsf{pub}(B)}$$

- $$B \to A : \{N_a, N_b\}_{\mathsf{pub}(A)}$$

$$A \to B : \{N_b\}_{\mathsf{pub}(B)}$$

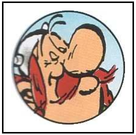


Needham-Schroeder's Protocol (1978)

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

Needham-Schroeder's Protocol (1978)

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



Needham-Schroeder's Protocol (1978)

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

Questions

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

Needham-Schroeder's Protocol (1978)

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

Questions

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

Attack

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

How cryptographic protocols can be attacked?

Breaking encryption

Verification of cryptographic protocols

How cryptographic protocols can be attacked?

Breaking encryption

Logical attack

Verification of cryptographic protocols

How cryptographic protocols can be attacked?

Breaking encryption

Logical attack

Logical attacks

- can be mounted even assuming perfect cryptography,
  ↪ replay attack, man-in-the middle attack, . . .
- are numerous, see SPORE, Security Protocols Open REpository
  ↪ `http://www.lsv.ens-cachan.fr/spore/`
- subtle and hard to detect by “eyeballing” the protocol

Example: Man in the Middle Attack

Agent A

Intrus I

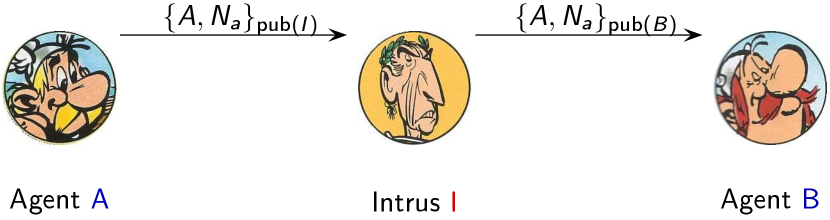
Agent B

Attack

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

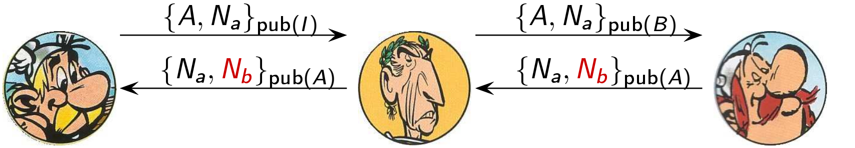
$$\begin{array}{lcl} A \rightarrow B & : & \{A, N_a\}_{\mathsf{pub}(B)} \\ B \rightarrow A & : & \{N_a, N_b\}_{\mathsf{pub}(A)} \\ A \rightarrow B & : & \{N_b\}_{\mathsf{pub}(B)} \end{array}$$

Example: Man in the Middle Attack



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

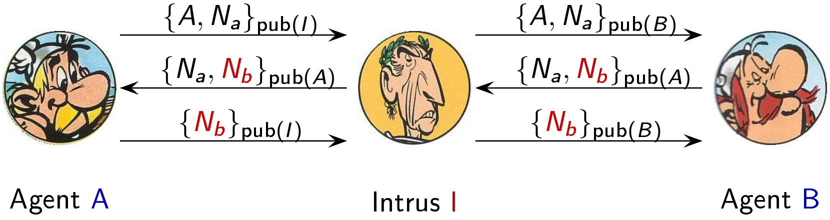
Example: Man in the Middle Attack



Agent A | Intrus I | Agent B

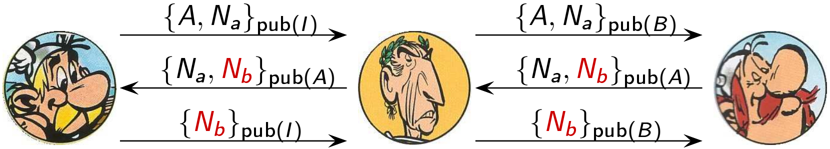
$$
\begin{array}{lll}
A \rightarrow B & : & \{A, N_a\}_{\mathrm{pub}(B)} \\
B \rightarrow A & : & \{N_a, N_b\}_{\mathrm{pub}(A)} \\
A \rightarrow B & : & \{N_b\}_{\mathrm{pub}(B)}
\end{array}
$$

Example: Man in the Middle Attack



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

Example: Man in the Middle Attack



Agent A | Intrus I | Agent B

Attack

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

$$A \rightarrow B \quad : \{A, N_a\}_{pub(B)}$$
$$B \rightarrow A \quad : \{N_a, N_b\}_{pub(A)}$$
$$A \rightarrow B \quad : \{N_b\}_{pub(B)}$$

Logical attacks - How to detect them?

Symbolic approach

- messages are represented by terms rather than bit-strings
  $\hookrightarrow$ $\{m\}_k$ encryption of the message m with key k,
  $\hookrightarrow$ $\langle m_1, m_2 \rangle$ pairing of messages m_1 and m_2, ...
- attacker controls the network and can perform specific actions

Logical attacks - How to detect them?

Symbolic approach

- messages are represented by terms rather than bit-strings
  $\hookrightarrow$ $\{m\}_k$ encryption of the message m with key k,
  $\hookrightarrow$ $\langle m_1, m_2 \rangle$ pairing of messages m_1 and m_2, ...
- attacker controls the network and can perform specific actions

Relevance of the approach

- numerous attacks have already been obtained,
- allows us to perform automatic verification, *e.g.* AVISPA, Proverif, ...
- soundness results already exist, *e.g.* [Micciancio & Warinschi'04]

Difficulties of the verification

Presence of an attacker ...

Difficulties of the verification

Presence of an attacker ...

who controls the communication network:

- may read every message sent on the network
- may intercept and send new messages

Difficulties of the verification

Presence of an attacker ...

who controls the communication network:

- may read every message sent on the network
- may intercept and send new messages

who has deduction capabilities (*e.g.* the standard Dolev-Yao model)

- encryption, decryption if he knows the decryption key,
- pairing, projection

Difficulties of the verification

Presence of an attacker ...

who controls the communication network:

- may read every message sent on the network
- may intercept and send new messages

who has deduction capabilities (*e.g.* the standard Dolev-Yao model)

- encryption, decryption if he knows the decryption key,
- pairing, projection

Security problem for an unbounded number of sessions is undecidable.

Difficulties of the verification

Presence of an attacker ...

who controls the communication network:

- may read every message sent on the network
- may intercept and send new messages

who has deduction capabilities (*e.g.* the standard Dolev-Yao model)

- encryption, decryption if he knows the decryption key,
- pairing, projection

Security problem for a fixed number of sessions is decidable.

Outline of the talk

1. Introduction
2. How to deal with trace properties? (e.g. secrecy, authentication)
3. Work in progress: Equivalence based security properties (e.g. anonymity)

Outline of the talk

Dolev-Yao Intruder Model

m_1, m_2 and A are messages (terms)
T a finite set of messages (intruder's knowledge)

Ax. (A) $$\frac{}{T \vdash m_1} \quad m_1 \in T$$

Pair (P) $$\frac{T \vdash m_1 \quad T \vdash m_2}{T \vdash \langle m_1, m_2 \rangle}$$

Enc. (E) $$\frac{T \vdash m_1 \quad T \vdash \mathsf{pub}(A)}{T \vdash \{m_1\}_{\mathsf{pub}(A)}}$$

Proj. (Prj$_2$) $$\frac{T \vdash \langle m_1, m_2 \rangle}{T \vdash m_2}$$

Dec. (D) $$\frac{T \vdash \{m_1\}_{\mathsf{pub}(A)} \quad T \vdash \mathsf{priv}(A)}{T \vdash m_1}$$

Proj. (Prj$_1$) $$\frac{T \vdash \langle m_1, m_2 \rangle}{T \vdash m_1}$$

Deducibility problem

Deducibility problem

INPUT: an intruder inference system $\mathcal{I}$, a finite set of terms T, a term s (the secret).

OUTPUT: Does there exist a proof of $T \vdash s$?

Deducibility problem

Deducibility problem

INPUT: an intruder inference system $\mathcal{I}$, a finite set of terms T, a term s (the secret).

OUTPUT: Does there exist a proof of $T \vdash s$?

Example: Is $\langle s_1, s_2 \rangle$ deducible from the set of terms T which contains s_1, $\{s_2\}_k$ and k?

Deducibility problem

Deducibility problem

INPUT: an intruder inference system $\mathcal{I}$, a finite set of terms T, a term s (the secret).

OUTPUT: Does there exist a proof of $T \vdash s$?

Example: Is $\langle s_1, s_2 \rangle$ deducible from the set of terms T which contains s_1, $\{s_2\}_k$ and k?

$$\dfrac{\dfrac{s_1 \in T}{T \vdash s_1}\,(A) \qquad \dfrac{\dfrac{\{s_2\}_k \in T}{T \vdash \{s_2\}_k}\,(A) \quad \dfrac{k \in T}{T \vdash k}\,(A)}{T \vdash s_2}\,(D)}{T \vdash \langle s_1, s_2 \rangle}\,(P)$$

Deducibility problem - Some existing results

$\longrightarrow$ depends on the deduction capabilities of the intruder

Dolev-Yao intruder

The deducibility problem is decidable in polynomial time.

Deducibility problem - Some existing results

$\longrightarrow$ depends on the deduction capabilities of the intruder

Dolev-Yao intruder

The deducibility problem is decidable in polynomial time.

Prefix Intruder (*e.g.* Cipher Block Chaining)

$$\frac{T \vdash \{\langle m_1, m_2 \rangle\}_{\mathsf{pub}(A)}}{T \vdash \{m_1\}_{\mathsf{pub}(A)}}$$

Deducibility problem - Some existing results

$\longrightarrow$ depends on the deduction capabilities of the intruder

Dolev-Yao intruder

The deducibility problem is decidable in polynomial time.

Prefix Intruder (*e.g.* Cipher Block Chaining)

$$\frac{T \vdash \{\langle m_1, m_2 \rangle\}_{\mathsf{pub}(A)}}{T \vdash \{m_1\}_{\mathsf{pub}(A)}}$$

Taking into account algebraic properties of the cryptographic primitives (*e.g.* RSA encrytpion)

$$\mathsf{E} := \begin{cases} \mathsf{dec}(\mathsf{enc}(x, \mathsf{pub}(y)), \mathsf{priv}(y)) & = & x \\ \mathsf{enc}(\mathsf{dec}(x, \mathsf{priv}(y)), \mathsf{pub}(y)) & = & x \end{cases}$$

$$\frac{T \vdash m \quad T \vdash k}{T \vdash \mathsf{f}(m, k)} \; \mathsf{f} \in \{\mathsf{dec}, \mathsf{enc}\} \qquad \frac{T \vdash m_1}{T \vdash m_2} \; m_1 =_{\mathsf{E}} m_2$$

In presence of an active attacker

Insecurity problem (bounded number of sessions)

Let $\mathcal{I}$ be an inference system modelling the attacker.

INPUT: a finite set $R_1, \ldots, R_m$ of instances of roles,
a finite set T_0 of terms (initial intruder knowledge),
a term s (the secret)

In presence of an active attacker

Insecurity problem (bounded number of sessions)

Let $\mathcal{I}$ be an inference system modelling the attacker.

INPUT: a finite set $R_1, \ldots, R_m$ of instances of roles,
a finite set T_0 of terms (initial intruder knowledge),
a term s (the secret)

OUTPUT: Does there exist an interleaving of $R_1, \ldots, R_m$ runnable from T_0 w.r.t. $\mathcal{I}$ at the end of which

- the intruder knowledge is T, and
- s is deducible from T in $\mathcal{I}$?

In presence of an active attacker

Insecurity problem (bounded number of sessions)

Let $\mathcal{I}$ be an inference system modelling the attacker.

INPUT: a finite set $R_1, \ldots, R_m$ of instances of roles,
a finite set T_0 of terms (initial intruder knowledge),
a term s (the secret)

OUTPUT: Does there exist an interleaving of $R_1, \ldots, R_m$ runnable from T_0 w.r.t. $\mathcal{I}$ at the end of which

- the intruder knowledge is T, and
- s is deducible from T in $\mathcal{I}$?

Security properties (trace properties): *e.g.* secrecy, some kinds of authentication properties, . . .

Running example: Needham-Schroeder's protocol

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

Roles composing the protocol

$$
\begin{array}{lcl}
R_A(x_a, x_b) & : & \nu n_a.\ \mathsf{out}(\{x_a, n_a\}_{\mathsf{pub}(x_b)}); \\
& & \quad \mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(x_a)}); \mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(x_b)}) \\
& & \\
R_B(y_b) & : & \nu n_b.\ \mathsf{in}(\{y_a, y_{n_a}\}_{\mathsf{pub}(y_b)}); \mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)})
\end{array}
$$

Running example: Needham-Schroeder's protocol

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

Roles composing the protocol

$$\begin{array}{lcl} R_A(x_a, x_b) & : & \nu n_a.\ \mathrm{out}(\{x_a, n_a\}_{\mathrm{pub}(x_b)}); \\ & & \quad \mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(x_a)}); \mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(x_b)}) \\ R_B(y_b) & : & \nu n_b.\ \mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(y_b)}); \mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)}) \end{array}$$

To retrieve the well-known man-in-the-middle attack, we consider

- $R_A(a, I)$ and $R_B(b)$ (running in parallel).
- $T_0 = \{a, b, I, \mathrm{pub}(a), \mathrm{pub}(b), \mathrm{pub}(I), \mathrm{priv}(I)\}$
- Is n_b deducible by the intruder?

Insecurity problem via constraint solving

Protocol rules	Constraint System

$$\begin{array}{c} \mathsf{in}(u_1);\ \mathsf{out}(v_1) \\ \mathsf{in}(u_2);\ \mathsf{out}(v_2) \\ \ldots \\ \mathsf{in}(u_n);\ \mathsf{out}(v_n) \end{array}$$

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

Insecurity problem via constraint solving

Protocol rules

$$\begin{array}{c} \mathrm{in}(u_1);\ \mathrm{out}(v_1) \\ \mathrm{in}(u_2);\ \mathrm{out}(v_2) \\ \ldots \\ \mathrm{in}(u_n);\ \mathrm{out}(v_n) \end{array}$$

Constraint System

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

Solution of a constraint system in $\mathcal{I}$

A substitution σ such that

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

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

$$\begin{array}{rcl} & & \mathrm{out}(\{a, n_a\}_{\mathrm{pub}(I)}) \\ \mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)}) & ; & \mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(I)}) \\ \\ \mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(b)}) & ; & \mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)}) \end{array}$$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(I)})$
3	$\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})$	;	$\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(I)})$
2	$\mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(b)})$	;	$\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)})$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(I)})$
3	$\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})$	;	$\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(I)})$
2	$\mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(b)})$	;	$\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)})$

Constraints System

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathsf{out}(\{a, n_a\}_{\mathsf{pub}(I)})$
3	$\mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)})$	;	$\mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(I)})$
2	$\mathsf{in}(\{y_a, y_{n_a}\}_{\mathsf{pub}(b)})$	;	$\mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)})$

Constraints System

$$T_0, \ \{a, n_a\}_{\mathsf{pub}(I)}$$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(I)})$
3	$\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})$	;	$\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(I)})$
2	$\mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(b)})$	;	$\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\mathrm{pub}(b)}$$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathsf{out}(\{a, n_a\}_{\mathsf{pub}(I)})$
3	$\mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)})$	;	$\mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(I)})$
2	$\mathsf{in}(\{y_a, y_{n_a}\}_{\mathsf{pub}(b)})$	;	$\mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\mathsf{pub}(b)}$$
$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)}$$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(I)})$
3	$\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})$	;	$\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(I)})$
2	$\mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(b)})$	;	$\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\mathrm{pub}(b)}$$
$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)} \Vdash \{n_a, x_{n_b}\}_{\mathrm{pub}(a)}$$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathsf{out}(\{a, n_a\}_{\mathsf{pub}(I)})$
3	$\mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)})$	;	$\mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(I)})$
2	$\mathsf{in}(\{y_a, y_{n_a}\}_{\mathsf{pub}(b)})$	;	$\mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\mathsf{pub}(b)}$$
$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)} \Vdash \{n_a, x_{n_b}\}_{\mathsf{pub}(a)}$$
$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)},\ \{x_{n_b}\}_{\mathsf{pub}(I)}$$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(I)})$
3	$\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})$	;	$\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(I)})$
2	$\mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(b)})$	;	$\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\mathrm{pub}(b)}$$
$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)} \Vdash \{n_a, x_{n_b}\}_{\mathrm{pub}(a)}$$
$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)},\ \{x_{n_b}\}_{\mathrm{pub}(I)} \Vdash n_b$$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathsf{out}(\{a, n_a\}_{\mathsf{pub}(I)})$
3	$\mathsf{in}(\{n_a, x_{n_b}\}_{\mathsf{pub}(a)})$	;	$\mathsf{out}(\{x_{n_b}\}_{\mathsf{pub}(I)})$
2	$\mathsf{in}(\{y_a, y_{n_a}\}_{\mathsf{pub}(b)})$	;	$\mathsf{out}(\{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\mathsf{pub}(b)}$$
$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)} \Vdash \{n_a, x_{n_b}\}_{\mathsf{pub}(a)}$$
$$T_0,\ \{a, n_a\}_{\mathsf{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathsf{pub}(y_a)},\ \{x_{n_b}\}_{\mathsf{pub}(I)} \Vdash n_b$$

Solution $\quad \sigma = \{y_a \mapsto \quad,\ y_{n_a} \mapsto \quad,\ x_{n_b} \mapsto \quad\}$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\text{out}(\{a, n_a\}_{\text{pub}(I)})$
3	$\text{in}(\{n_a, x_{n_b}\}_{\text{pub}(a)})$	;	$\text{out}(\{x_{n_b}\}_{\text{pub}(I)})$
2	$\text{in}(\{y_a, y_{n_a}\}_{\text{pub}(b)})$	;	$\text{out}(\{y_{n_a}, n_b\}_{\text{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\text{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\text{pub}(b)}$$
$$T_0,\ \{a, n_a\}_{\text{pub}(I)},\ \{y_{n_a}, n_b\}_{\text{pub}(y_a)} \Vdash \{n_a, x_{n_b}\}_{\text{pub}(a)}$$
$$T_0,\ \{a, n_a\}_{\text{pub}(I)},\ \{y_{n_a}, n_b\}_{\text{pub}(y_a)},\ \{x_{n_b}\}_{\text{pub}(I)} \Vdash n_b$$

Solution $\sigma = \{y_a \mapsto a,\ \ y_{n_a} \mapsto n_a,\ \ x_{n_b} \mapsto \quad \}$

Running example: Needham-Schroeder's protocols

$R_A(a, I)$ and $R_B(b)$ (running in parallel)

1			$\mathrm{out}(\{a, n_a\}_{\mathrm{pub}(I)})$
3	$\mathrm{in}(\{n_a, x_{n_b}\}_{\mathrm{pub}(a)})$	;	$\mathrm{out}(\{x_{n_b}\}_{\mathrm{pub}(I)})$
2	$\mathrm{in}(\{y_a, y_{n_a}\}_{\mathrm{pub}(b)})$	;	$\mathrm{out}(\{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)})$

Constraints System

$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)} \Vdash \{y_a, y_{n_a}\}_{\mathrm{pub}(b)}$$
$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)} \Vdash \{n_a, x_{n_b}\}_{\mathrm{pub}(a)}$$
$$T_0,\ \{a, n_a\}_{\mathrm{pub}(I)},\ \{y_{n_a}, n_b\}_{\mathrm{pub}(y_a)},\ \{x_{n_b}\}_{\mathrm{pub}(I)} \Vdash n_b$$

Solution $\quad \sigma = \{y_a \mapsto a,\ \ y_{n_a} \mapsto n_a,\ \ x_{n_b} \mapsto n_b\}$

Existing results

Many theoretical results for different intruder models

- to take into account algebraic properties of cryptographic primitives (exclusive or, cipher block chaining, ...)
- to take into account the fact that some data are poorly-chosen (*e.g.* passwords)

Existing results

Many theoretical results for different intruder models

- to take into account algebraic properties of cryptographic primitives (exclusive or, cipher block chaining, ...)
- to take into account the fact that some data are poorly-chosen (*e.g.* passwords)

Few generic results

- procedure to solve constraint systems for a class of intruder
  $\hookrightarrow$ *e.g.* any intruder who can be described by a subterm convergent rewiting system
- combination result for disjoint intruder models.

Existing results

Many theoretical results for different intruder models

- to take into account algebraic properties of cryptographic primitives (exclusive or, cipher block chaining, ...)
- to take into account the fact that some data are poorly-chosen (*e.g.* passwords)

Few generic results

- procedure to solve constraint systems for a class of intruder
  $\hookrightarrow$ *e.g.* any intruder who can be described by a subterm convergent rewiting system
- combination result for disjoint intruder models.

Some tools

- AVISPA tool (Atse, OFMC)

Outline of the talk

1. Introduction
2. How to deal with trace properties? (e.g. secrecy, authentication)
3. Work in progress: Equivalence based security properties (e.g. anonymity)

Motivation: Electronic voting

Advantages:

- Convenient,
- Efficient facilities for tallying votes.

Drawbacks:

- Risk of large-scale and undetectable fraud,
- Such protocols are extremely error-prone.

> *"A 15-year-old in a garage could manufacture smart cards and sell them on the Internet that would allow for multiple votes"* Avi Rubin

Possible issue: formal methods

abstract analysis of the protocol against formally-stated properties

Expected properties

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

Receipt-freeness: a voter cannot prove that she voted in a certain way (this is important to protect voters from coercion)

Coercion-resistance: same as receipt-freeness, but the coercer interacts with the voter during the protocol, e.g. by preparing messages

How to model such security properties?

[Kremer & Ryan, 2005] – Formalisation of Privacy
$\hookrightarrow$ consider 2 honest voters and swap their votes

Privacy

A voting protocol respects privacy if

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx S[V_A\{^b/_v\} \mid V_B\{^a/_v\}].$$

[Delaune, Kremer & Ryan, 2006]
Formalisation of Receipt-freeness and Coercion-resistance in term of equivalence.

Some examples

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx S[V_A\{^b/_v\} \mid V_B\{^a/_v\}]$$

Naive vote protocol (version 1)

$$V \rightarrow S : \{v\}_{\text{pub}(S)}$$

What about privacy?

Some examples

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx S[V_A\{^b/_v\} \mid V_B\{^a/_v\}]$$

Naive vote protocol (version 1)

$$V \rightarrow S : \{v\}_{\mathsf{pub}(S)}$$

What about privacy? OK

Some examples

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx S[V_A\{^b/_v\} \mid V_B\{^a/_v\}]$$

Naive vote protocol (version 1)

$$V \rightarrow S : \{v\}_{\mathsf{pub}(S)}$$

What about privacy? OK

Naive vote protocol (version 2)

$$V \rightarrow S : Id, \{v\}_{\mathsf{pub}(S)}$$

What about privacy?

Some examples

$$S[V_A\{^a/_v\} \mid V_B\{^b/_v\}] \approx S[V_A\{^b/_v\} \mid V_B\{^a/_v\}]$$

Naive vote protocol (version 1)

$$V \rightarrow S : \{v\}_{\mathsf{pub}(S)}$$

What about privacy? OK

Naive vote protocol (version 2)

$$V \rightarrow S : Id, \{v\}_{\mathsf{pub}(S)}$$

What about privacy?

- deterministic encryption: NOT OK
- probabilistic encryption: OK

More formally

Labeled bisimilarity ($\approx_\ell$)

The largest symmetric relation $\mathcal{R}$ on processes, such that $A \mathcal{R} B$ implies

1. $\phi(A) \approx_s \phi(B)$ (depends on E),
2. if $A \rightarrow A'$, then $B \rightarrow^* B'$ and $A' \mathcal{R} B'$ for some B',
3. if $A \xrightarrow{\alpha} A'$, then $B \rightarrow^* \xrightarrow{\alpha} \rightarrow^* B'$ and $A' \mathcal{R} B'$ for some B'.

This relation is in genral undecidable. Why?

- unfolding tree is infinite in depth
- unfolding tree is infinititely branching (because of inputs)
- equational theories may be complex

Tool: Proverif

$\longrightarrow$ Obviously, the procedure is not complete. Proverif is not able to conclude for privacy even for naive voting protocols (version 1)

Work in Progress

Our Goal:

to do better than Proverif in the context of a bounded number of sessions

- Infinite depth:
  $\hookrightarrow$ we restrict to consider processes without replication (finite processes),
- Infinite branching:
  $\hookrightarrow$ we define a notion of symbolic processes and symbolic bisimulation

Concrete	$in(x).out(\{x\}_k) \xrightarrow{in(m_1)} out(\{m_1\}_k)$
Symbolic	$(in(x).out(\{x\}_k);\, \mathcal{C}) \xrightarrow{in(x)} (out(\{x\}_k);\, \mathcal{C} \cup \phi(P) \Vdash x)$

Work in Progress

Our Goal:

to do better than Proverif in the context of a bounded number of sessions

- Infinite depth:
  $\hookrightarrow$ we restrict to consider processes without replication (finite processes),
- Infinite branching:
  $\hookrightarrow$ we define a notion of symbolic processes and symbolic bisimulation

Concrete $\quad in(x).out(\{x\}_k) \xrightarrow{in(m_1)} out(\{m_1\}_k)$

Symbolic $\quad (in(x).out(\{x\}_k);\ \mathcal{C}) \xrightarrow{in(x)} (out(\{x\}_k);\ \mathcal{C} \cup \phi(P) \Vdash x)$

Then, we plan:

- to design a procedure to solve our constaint systems for a class of equational theory as larger as possible
- to implement a tool