

# Security via constraint solving

Stéphanie Delaune

October 30, 2006

## Cryptographic protocols

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



[cliquer ici pour accéder à la signature de votre déclaration](#)

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

- **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)



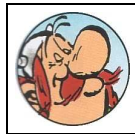
- $A \rightarrow B : \{A, N_a\}_{\text{pub}(B)}$   
 $B \rightarrow A : \{N_a, N_b\}_{\text{pub}(A)}$   
 $A \rightarrow B : \{N_b\}_{\text{pub}(B)}$



# Needham-Schroeder's Protocol (1978)



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



# Needham-Schroeder's Protocol (1978)



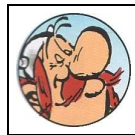
- $A \rightarrow B : \{A, N_a\}_{\text{pub}(B)}$   
 $B \rightarrow A : \{N_a, N_b\}_{\text{pub}(A)}$   
 $A \rightarrow B : \{N_b\}_{\text{pub}(B)}$



# Needham-Schroeder's Protocol (1978)



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



# Needham-Schroeder's Protocol (1978)



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



## Questions

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

# Needham-Schroeder's Protocol (1978)



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



## 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]

# Verification of cryptographic protocols

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



Intruder I



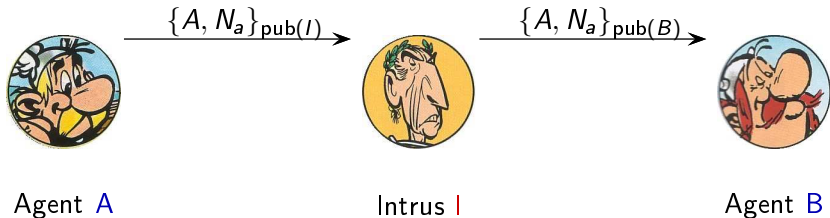
Agent B

## Attack

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

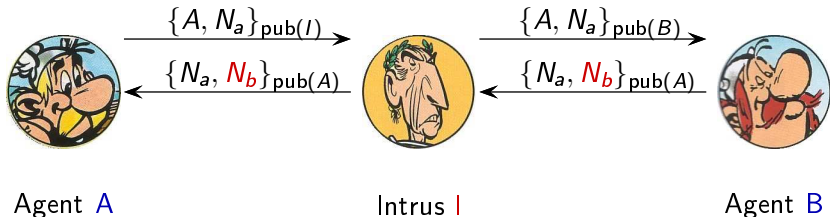
$$\begin{aligned} A \rightarrow B & : \{A, N_a\}_{\text{pub}(B)} \\ B \rightarrow A & : \{N_a, N_b\}_{\text{pub}(A)} \\ A \rightarrow B & : \{N_b\}_{\text{pub}(B)} \end{aligned}$$

# Example: Man in the Middle Attack



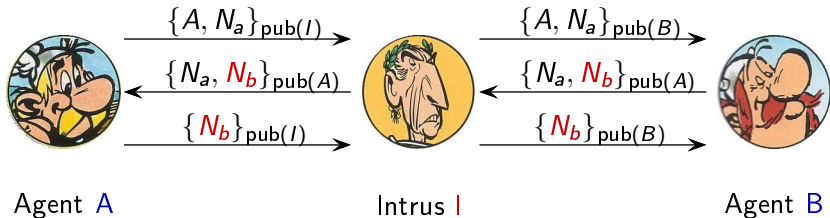
$A \rightarrow B : \{A, N_a\}_{\text{pub}(B)}$   
 $B \rightarrow A : \{N_a, N_b\}_{\text{pub}(A)}$   
 $A \rightarrow B : \{N_b\}_{\text{pub}(B)}$

# Example: Man in the Middle Attack



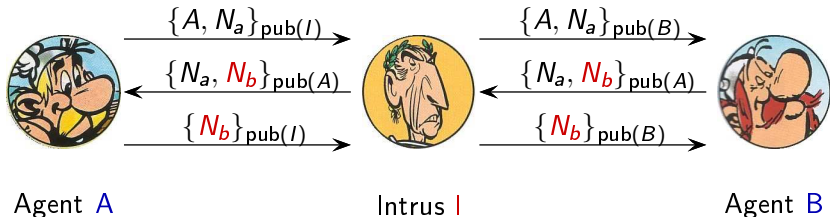
$A \rightarrow B : \{A, N_a\}_{\text{pub}(B)}$   
 $B \rightarrow A : \{N_a, N_b\}_{\text{pub}(A)}$   
 $A \rightarrow B : \{N_b\}_{\text{pub}(B)}$

# Example: Man in the Middle Attack



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

# Example: Man in the Middle Attack



## Attack

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

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

## Symbolic approach

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

## Relevance of the approach

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

# 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

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

# Dolev-Yao Intruder Model



$m_1, m_2$  and  $A$  are messages (terms)

$T$  a finite set of messages (intruder's knowledge)

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

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

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

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

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

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

## 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  $k$ ?

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

## Deducibility problem - Some existing results

→ depends on the deduction capabilities of the intruder

### Dolev-Yao intruder

The deducibility problem is decidable in **polynomial time**.

# Deducibility problem - Some existing results

→ 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)

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

# Deducibility problem - Some existing results

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

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

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

$$\frac{T \vdash m \quad T \vdash k}{T \vdash f(m, k)} \quad f \in \{\text{dec}, \text{enc}\} \quad \frac{T \vdash m_1}{T \vdash m_2} \quad m_1 =_E m_2$$

## Insecurity problem (bounded number of sessions)

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

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

## Insecurity problem (bounded number of sessions)

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

INPUT: a finite set  $R_1, \dots, R_m$  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  $R_1, \dots, R_m$   
**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  $\mathcal{I}$ ?

## Insecurity problem (bounded number of sessions)

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

INPUT: a finite set  $R_1, \dots, R_m$  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  $R_1, \dots, R_m$   
**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

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

## Roles composing the protocol

$$R_A(x_a, x_b) : \nu n_a. \text{ out}(\{x_a, n_a\}_{\text{pub}(x_b)}); \\ \text{ in}(\{n_a, x_{n_b}\}_{\text{pub}(x_a)}); \text{ out}(\{x_{n_b}\}_{\text{pub}(x_b)})$$
$$R_B(y_b) : \nu n_b. \text{ in}(\{y_a, y_{n_a}\}_{\text{pub}(y_b)}); \text{ out}(\{y_{n_a}, n_b\}_{\text{pub}(y_a)})$$

# Running example: Needham-Schroeder's protocol

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

## Roles composing the protocol

$$R_A(x_a, x_b) : \nu n_a. \text{ out}(\{x_a, n_a\}_{\text{pub}(x_b)}); \\ \text{ in}(\{n_a, x_{n_b}\}_{\text{pub}(x_a)}); \text{ out}(\{x_{n_b}\}_{\text{pub}(x_b)})$$
$$R_B(y_b) : \nu n_b. \text{ in}(\{y_a, y_{n_a}\}_{\text{pub}(y_b)}); \text{ out}(\{y_{n_a}, n_b\}_{\text{pub}(y_a)})$$

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

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

# Insecurity problem via constraint solving

Protocol rules

in( $u_1$ ); out( $v_1$ )  
in( $u_2$ ); out( $v_2$ )  
...  
in( $u_n$ ); out( $v_n$ )

Constraint System

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

Protocol rules

$\text{in}(u_1); \text{out}(v_1)$   
 $\text{in}(u_2); \text{out}(v_2)$   
...  
 $\text{in}(u_n); \text{out}(v_n)$

Constraint System

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

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

A substitution  $\sigma$  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)

$\text{in}(\{n_a, x_{n_b}\}_{\text{pub}(a)})$  ;  $\text{out}(\{a, n_a\}_{\text{pub}(I)})$   
 $\text{out}(\{x_{n_b}\}_{\text{pub}(I)})$

$\text{in}(\{y_a, y_{n_a}\}_{\text{pub}(b)})$  ;  $\text{out}(\{y_{n_a}, n_b\}_{\text{pub}(y_a)})$

# Running example: Needham-Schroeder's protocols

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

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



# Running example: Needham-Schroeder's protocols

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

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

Constraints System

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

# Running example: Needham-Schroeder's protocols

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

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

Constraints System

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

# Running example: Needham-Schroeder's protocols

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

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

Constraints System

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

# Running example: Needham-Schroeder's protocols

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

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

Constraints System

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

# Running example: Needham-Schroeder's protocols

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

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

Constraints System

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

# Running example: Needham-Schroeder's protocols

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

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

Constraints System

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

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

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

## 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)

## 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  
↔ e.g. any intruder who can be described by a subterm convergent rewriting system
- **combination** result for **disjoint** intruder models.

## 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  
↔ e.g. any intruder who can be described by a subterm convergent rewriting 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

**Privacy:** the fact that a particular 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**

↪ 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)

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

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)

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

What about privacy? **OK**

### Naive vote protocol (version 2)

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

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)

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

What about privacy? **OK**

### Naive vote protocol (version 2)

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

What about privacy?

- **deterministic** encryption: **NOT OK**
- **probabilistic** encryption: **OK**

## Labeled bisimilarity ( $\approx_l$ )

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

- 1  $\phi(A) \approx_s \phi(B)$  (depends on  $\mathbf{E}$ ),
- 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 general **undecidable**. Why?

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

Tool: **Proverif**

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

## Our Goal:

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

- **Infinite depth:**

↔ we restrict to consider processes without replication (finite processes),

- **Infinite branching:**

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

## Our Goal:

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

- **Infinite depth:**

↔ we restrict to consider processes without replication (finite processes),

- **Infinite branching:**

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

Then, we plan:

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