

# **Models and analysis of security protocols**

## **1st Semester 2007-2008**

### **Dolev Yao and Passive Intruder**

**Pascal Lafourcade**

*Université Joseph Fourier, Verimag*

Master : October 18th 2007

# Last Time (I)

## Lecture

- Presentation
- Motivation
- Cryptography
- Logical Attacks
- Needham Schroeder

## Last Time (I)

### Exercise

- Airport Security
- Needham Schroeder
- Cryptography

# Outline of Today

- 1 Dolev Yao's Intruder
- 2 Terms and Messages
- 3 Notion of Locality
- 4 Passive Intruder: Intruder Deduction Problem
- 5 Undecidability for unbounded number of sessions
- 6 Conclusion

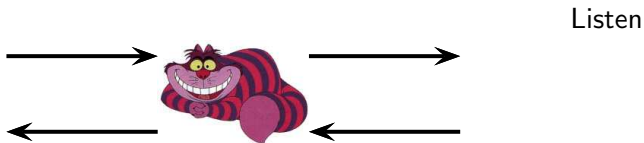
# Outline

- 1 Dolev Yao's Intruder
- 2 Terms and Messages
- 3 Notion of Locality
- 4 Passive Intruder: Intruder Deduction Problem
- 5 Undecidability for unbounded number of sessions
- 6 Conclusion

## The Intruder is the Network (Worst Case)



## The Intruder is the Network (Worst Case)



Passive: Intruder deduction problem

## The Intruder is the Network (Worst Case)



Listen

Intercept message

(Re)play message

Delete message

Passive: Intruder deduction problem

Active: Security problem

## The Intruder is the Network (Worst Case)



Listen

Intercept message

(Re)play message

Delete message

Passive: Intruder deduction problem

Active: Security problem

### Intruder Capabilities (Dolev-Yao Model 80's)

- Encryption, Decryption with a key
- Pairing, Projection.

## Proof System

A **sequent** is an expression of the form  $T \vdash u$ .

### Definition

A **proof** of a sequent  $T \vdash u$  is a tree whose nodes are labeled by either sequents or expressions of the form " $v \in T$ ", such that:

- Each leaf is labeled by an expression of the form  $v \in T$ , and each non-leaf node is labeled by an sequent.
- Each node labeled by a sequent  $T \vdash v$  has  $n$  children labeled by  $T \vdash s_1, \dots, T \vdash s_n$  such that there is an instance of an inference rule with conclusion  $T \vdash_E v$  and **hypotheses**  $T \vdash s_1, \dots, T \vdash s_n$ .
- The **root** of the tree is labeled by  $T \vdash u$ .

A **subproof** of a proof  $P$  is a subtree of  $P$ .

## Notions for Proof System

### Definition

- **Size of a proof**  $P$  of  $T \vdash u$  is denoted by  $|P|$ , is the number of nodes in the proof.
- A proof  $P$  of  $T \vdash u$  is **minimal** if there does not exist a proof  $P'$  of  $T \vdash u$  such that  $|P'| < |P|$ .

## Dolev-Yao Deduction System

Deduction System :  $T_0 \vdash^? s$

$$(A) \quad \frac{u \in T_0}{T_0 \vdash u}$$

$$(UL) \quad \frac{T_0 \vdash \langle u, v \rangle}{T_0 \vdash u}$$

$$(P) \quad \frac{T_0 \vdash u \quad T_0 \vdash v}{T_0 \vdash \langle u, v \rangle}$$

$$(UR) \quad \frac{T_0 \vdash \langle u, v \rangle}{T_0 \vdash v}$$

$$(C) \quad \frac{T_0 \vdash u \quad T_0 \vdash v}{T_0 \vdash \{u\}_v}$$

$$(D) \quad \frac{T_0 \vdash \{u\}_v \quad T_0 \vdash v}{T_0 \vdash u}$$

Example:  $T_0 \vdash^? s$

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = b$

Example:  $T_0 \vdash^? s$

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = b$

$$\begin{array}{c}
 \begin{array}{c}
 (A) \frac{\{b\}_c \in T_0}{T_0 \vdash \{b\}_c} \quad (D) \frac{\quad}{T_0 \vdash c} \\
 \hline
 (D) \frac{\quad}{T_0 \vdash b}
 \end{array}
 \quad
 \begin{array}{c}
 (A) \frac{T_0 \vdash \langle a, \{c\}_k \rangle \in T_0}{T_0 \vdash \langle a, \{c\}_k \rangle} \\
 (UR) \frac{\quad}{T_0 \vdash \{c\}_k} \\
 \hline
 (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array}
 \end{array}$$

## Dolev-Yao 1982

- Intruder controls the network and can:
  - intercept messages
  - modify messages
  - block messages
  - generate new messages
  - insert new messages
- Perfect cryptography:
  - Abstraction with terms algebra
  - Decryption only if inverse key is known
- Protocol has
  - Arbitrary number of principals
  - Arbitrary number of parallel sessions
  - Messages with arbitrary size

# Outline

- 1 Dolev Yao's Intruder
- 2 Terms and Messages**
- 3 Notion of Locality
- 4 Passive Intruder: Intruder Deduction Problem
- 5 Undecidability for unbounded number of sessions
- 6 Conclusion

# Arity

## Definition

- $\mathcal{F}$  is a finite set
  - *Arity* is a mapping from  $\mathcal{F}$  into  $\mathbb{N}$
  - $(\mathcal{F}, \textit{Arity})$  is a **ranked alphabet** or **signature** denoted  $\Sigma$
- 
- The **arity** of a symbol  $f \in \mathcal{F}$  is  $\textit{Arity}(f)$
  - The set of symbols of arity  $p$  is denoted by  $\mathcal{F}_p$ .
  - Elements of arity 0, 1,  $\dots$   $p$  are respectively called constants, unary,  $\dots$   $p$ -ary symbols.

# Example

## Example

Let  $\mathcal{F} = \{\mathbf{enc}, \mathbf{pair}, \mathbf{k}_1, \mathbf{k}_2, \mathbf{0}, \mathbf{1}\}$

$Ariety(\mathbf{enc}) = Ariety(\mathbf{pair}) = 2$

$Ariety(\mathbf{k}_1) = Ariety(\mathbf{k}_2) = Ariety(\mathbf{0}) = Ariety(\mathbf{1}) = 0$

We also denote  $\mathcal{F} = \{\mathbf{enc}/2, \mathbf{pair}/2, \mathbf{k}_1/0, \mathbf{k}_2/0, \mathbf{0}/0, \mathbf{1}/0\}$

# Terms

Let  $\mathcal{X}$  be a set of symbols called **variables**.

## Definition

The set  $\mathcal{T}(\mathcal{F}, \mathcal{X})$  of **terms** over the ranked alphabet  $\mathcal{F}$  and the set of variables  $\mathcal{X}$  is the smallest set defined by:

- $\mathcal{F}_0 \subseteq \mathcal{T}(\mathcal{F}, \mathcal{X})$
- $\mathcal{X} \subseteq \mathcal{T}(\mathcal{F}, \mathcal{X})$
- if  $p \geq 1$ ,  $f \in \mathcal{F}_p$  and  $t_1, \dots, t_p \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ , then  $f(t_1, \dots, t_p) \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ .

- If  $\mathcal{X} = \emptyset$  then  $\mathcal{T}(\mathcal{F}, \mathcal{X})$  is also written  $\mathcal{T}(\mathcal{F})$ . Terms in  $\mathcal{T}(\mathcal{F})$  are called **ground terms**.
- A term in  $\mathcal{T}(\mathcal{F}, \mathcal{X})$  is **linear** if each variable occurs at most once in  $t$ .

## Example

### Example

Let  $\mathcal{F} = \{\mathbf{enc}/2, \mathbf{pair}/2, \mathbf{k}_1/0, \mathbf{k}_2/0, \mathbf{0}/0, \mathbf{1}/0\}$  and  $\mathcal{X} = \{x, y, z\}$   
 $\mathbf{pair}(x, \mathbf{1})$ ,  $\mathbf{enc}(\mathbf{pair}(y, z), \mathbf{k}_1)$  and  $\mathbf{enc}(\mathbf{0}, \mathbf{k}_1)$  are terms in  $\mathcal{T}(\mathcal{F}, \mathcal{X})$   
 $\mathbf{pair}(\mathbf{0}, \mathbf{1})$ ,  $\mathbf{enc}(\mathbf{0}, \mathbf{k}_1)$  are terms in  $\mathcal{T}(\mathcal{F})$ , i.e., ground terms

We also denote  $\{-\}_-$  for  $\mathbf{enc}(-, -)$  and  $\langle -, - \rangle$  for  $\mathbf{pair}(-, -)$ .

# Term Representation

## Example

Let  $\mathcal{F} = \{cons(, ), nil, a\}$ . A term  $u = cons(a, cons(a, nil))$  is represented by:



- The set of variables occurring in a term  $t$  is denoted by  $vars(t)$ .

# Position

## Definition

The set of **positions**  $Pos(t)$  of one term  $t$  is defined inductively:

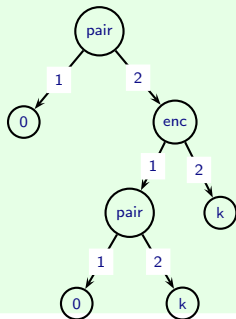
- $Pos(x) = Pos(n) = Pos(c) = \epsilon$
- $Pos(f(t_1, \dots, t_n)) = \{\epsilon\} \cup_{1 \leq i \leq n} i.Pos(t_i)$

## Example

$t = \mathbf{pair}(0, \mathbf{enc}(\mathbf{pair}(0, k), k))$

$Pos(t) = \{\epsilon, 1, 2, 21, 22, 211, 212\}$

The set of positions of a term  $t$  is also written  $\mathcal{O}(t)$ .



## Size

### Definition (Size of a term)

The **size of a term**  $t$ , denoted by  $|t|$  is:

$$\begin{aligned} |t| &= 1 \text{ if } t \in \mathcal{F}_0 \cup \mathcal{X} \\ |f(t_1, \dots, t_n)| &= 1 + \sum_{i=1}^n |t_i| \text{ if } f \in \mathcal{F}_n \end{aligned}$$

It is the number of nodes in  $t$ .

We extend this notion to set of terms

$$|\{t_1, \dots, t_n\}| = \sum_{i=1}^n |t_i|$$

## Definition of Syntactic Subterms

The **subterm**  $t|_p$  of  $t$  at the position  $p$  ( $p \in Pos(t)$ ) is:

$$t|_{\epsilon} = t \quad t|_{i \cdot p} = t_i|_p \text{ if } t = f(t_1, \dots, t_n), f \in \mathcal{F}_n$$

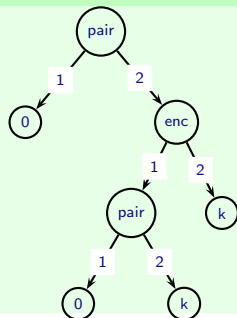
We denote  $S(t) = \{t|_p \mid p \in Pos(t)\}$  the set of subterms of  $t$ .

### Example

Let  $t = \mathbf{pair}(0, \mathbf{enc}(\mathbf{pair}(0, k), k))$

$t|_{21} = \mathbf{pair}(0, k)$

$S(t) = \{t, 0, \mathbf{enc}(\mathbf{pair}(0, k), k), \mathbf{pair}(0, k), k\}$



Extension to set of subterms is:  $S(\{t_1, \dots, t_n\}) = \cup_{1 \leq i \leq n} S(t_i)$

## Size DAG

### Definition (Size DAG of a term)

The **Size DAG of a term**  $t$ , denoted by  $|t|_{DAG}$  is the number of different subterms, i.e.,  $|t|_{DAG} = |S(t)|$  (where  $|E|$  is the cardinality of the set  $E$ ).

We extend this notion to set of terms

$$|\{t_1, \dots, t_n\}|_{DAG} = \left| \bigcup_{i=1}^n t_i \right|$$

- The term obtained by replacing  $t|_p$  with  $s$  is denoted  $t[s]_p$ .

# Syntactic Subterms

## Equivalent definition for Dolev Yao model

$S(t)$  is the smallest set such that:

- $t \in S(t)$
- $\langle u, v \rangle \in S(t) \Rightarrow u, v \in S(t)$
- $\{u\}_v \in S(t) \Rightarrow u, v \in S(t)$

Exercise:

- Let  $t = \{\langle a, \{b\}_{k_2} \rangle\}_{k_1}$
- Prove that the two definitions are equivalent.

## Syntactic Subterms

Equivalent definition for Dolev Yao model

$S(t)$  is the smallest set such that:

- $t \in S(t)$
- $\langle u, v \rangle \in S(t) \Rightarrow u, v \in S(t)$
- $\{u\}_v \in S(t) \Rightarrow u, v \in S(t)$

Exercise:

- Let  $t = \{\langle a, \{b\}_{k_2} \rangle\}_{k_1}$

$$S(t) = \{t, a, b, k_1, k_2, \{b\}_{k_2}, \langle a, \{b\}_{k_2} \rangle\}$$

- Prove that the two definitions are equivalent.

## Exercise on $S$

### Definition

$S(t)$  is the smallest set such that:

- $t \in S(t)$
- $\langle u, v \rangle \in S(t) \Rightarrow u, v \in S(t)$
- $\{u\}_v \in S(t) \Rightarrow u, v \in S(t)$

Let  $T$  be a set of terms.

- 1  $S(A \cup B) = S(A) \cup S(B)$
- 2  $S$  is idempotent:  $S(S(A)) = S(A)$
- 3  $S$  is monotonous: if  $A \subseteq B$  then  $S(A) \subseteq S(B)$
- 4  $S$  is transitive: if for all  $X, Y, Z \subseteq T$ ,  $X \subseteq S(Y)$  and  $Y \subseteq S(Z)$  implies  $X \subseteq S(Z)$ .

## Proof on $S$

- 1 Obvious from the way how the definition of  $S(T)$  is extended to sets.
- 2 Idempotence and Monotonicity  $\Rightarrow$  Transitivity:  
If  $X \subseteq S(Y)$  and  $Y \subseteq S(Z)$  by monotonicity and idempotence we get  $S(Y) \subseteq S(S(Z)) = S(Z)$ , hence  $X \subseteq S(Z)$ .

## Exercise on $S$

### Proposition

Let  $T$  be a set of terms then  $\#(S(T)) \leq |T|$ .

## proof

It is enough to show it on a term  $t$  since

$$\#(S(T)) = \#\left(\bigcup_{t \in T} S(t)\right) \leq \sum_{t \in T} \#(S(t)) \leq \sum_{t \in T} |t| = |T|$$

By induction on the size of the term  $t$ :

- The base case:  $t$  is a constant  $\#(S(t)) = 1 = |t|$ .

We analyze all possible cases to construct  $S(T)$ :

- $\#(S(\langle u, v \rangle)) \leq 1 + \#(S(u)) + \#(S(v)) \leq 1 + |u| + |v| = |\langle u, v \rangle|$
- $\#(S(\{u\}_v)) \leq 1 + \#(S(u)) + \#(S(v)) \leq 1 + |u| + |v| = |\{u\}_v|$

# Substitution

## Definition

- A **substitution** (respectively a **ground substitution**)  $\sigma$  is a mapping from  $\mathcal{X}$  into  $\mathcal{T}(\mathcal{F}, \mathcal{X})$  (respectively into  $\mathcal{T}(\mathcal{F})$ ) where there are only finitely many variables not mapped to themselves.
- Substitutions can be extended to  $\mathcal{T}(\mathcal{F}, \mathcal{X})$  in such a way that  $\forall f \in \mathcal{F}_n, \forall t_1, \dots, t_n \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ :

$$\sigma(f(t_1, \dots, t_n)) = f(\sigma(t_1), \dots, \sigma(t_n)).$$

The **domain** of a substitution  $\sigma$  is the subset of variables  $x \in \mathcal{X}$  such that  $\sigma(x) \neq x$ .

## Example:

Let  $\sigma = \{x \rightarrow N_A, y \rightarrow \{\langle N_A, N_B \rangle\}_{k_B}\}$  and  $t = \langle x, \langle y, \langle x, x \rangle \rangle \rangle$ .

Then,

$$\sigma(t) = \langle N_A, \{\{\langle N_A, N_B \rangle\}_{k_B}, \langle N_A, N_A \rangle\} \rangle$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

$$s = a \quad t = p(X)$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

$$s = a \quad t = p(X) \quad \text{No unifier}$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

$$s = a \quad t = p(X) \quad \text{No unifier}$$

$$s = p(a, X) \quad t = p(Y, b)$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

$$s = a \quad t = p(X) \quad \text{No unifier}$$

$$s = p(a, X) \quad t = p(Y, b) \quad \sigma = \{X \rightarrow b; Y \rightarrow a\}$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

$$s = a \quad t = p(X) \quad \text{No unifier}$$

$$s = p(a, X) \quad t = p(Y, b) \quad \sigma = \{X \rightarrow b; Y \rightarrow a\}$$

$$s = p(f(X), g(Z)) \quad t = p(f(a), Y)$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

$$s = a \quad t = p(X) \quad \text{No unifier}$$

$$s = p(a, X) \quad t = p(Y, b) \quad \sigma = \{X \rightarrow b; Y \rightarrow a\}$$

$$s = p(f(X), g(Z)) \quad t = p(f(a), Y) \\ \sigma = \{X \rightarrow a; Y \rightarrow g(Z)\}$$

# Unification

## Definition

Two  $t$  and  $u$  are unifiable if there exists a substitution  $\sigma$  such that  $\sigma s = \sigma t$

Examples:

$$s = a \quad t = X \quad \sigma = \{X \rightarrow a\}$$

$$s = a \quad t = p(X) \quad \text{No unifier}$$

$$s = p(a, X) \quad t = p(Y, b) \quad \sigma = \{X \rightarrow b; Y \rightarrow a\}$$

$$s = p(f(X), g(Z)) \quad t = p(f(a), Y)$$

$$\sigma = \{X \rightarrow a; Y \rightarrow g(Z)\} \text{ or } \sigma = \{X \rightarrow a; Y \rightarrow g(b); Z \rightarrow b\}$$

## Most General Unifier

### Definition

The most general unification between two terms  $s$  and  $t$ , denoted by  $mgu(s, t)$  if:  $\forall \sigma$  such that  $s\sigma = t\sigma, \exists \theta$  such that  $\sigma = mgu(s, t)\theta$

$$s = p(f(X), g(Z)) \quad t = p(f(a), Y)$$

$$\sigma_1 = \{X \rightarrow a; Y \rightarrow g(Z)\} \quad \sigma_2 = \{X \rightarrow a; Y \rightarrow g(b); Z \rightarrow b\}$$

# Goal

Design an algorithm that for a given unification problem  $s =? t$

- returns an mgu of  $s$  and  $t$  if they are unifiable.
- reports failure otherwise.

# Naive Algorithm

Write down two terms and set markers at the beginning of the terms. Then:

- 1 Move the markers simultaneously, one symbol at a time, until both move off the end of the term (success), or until they point to two different symbols;
- 2 If the two symbols are both non-variables, then fail; otherwise, one is a variable (call it  $x$ ) and the other one is the first symbol of a subterm (call it  $t$ ):
  - If  $x$  occurs in  $t$ , then fail;
  - Otherwise, replace  $x$  everywhere by  $t$  (including in the solution), write down " $x \rightarrow t$ " as a part of the solution, and return to 1.

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(x, g(a), g(z))$

$f(g(y), g(y), g(g(x)))$

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(x, g(a), g(z))$

$f(g(y), g(y), g(g(x)))$

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(x, g(a), g(z))$

$f(g(y), g(y), g(g(x)))$

$\sigma = \{x \rightarrow g(y)\}$

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(g(y), g(a), g(z))$

$f(g(y), g(y), g(g(g(y))))$

$\sigma = \{x \rightarrow g(y)\}$

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(g(y), g(a), g(z))$

$f(g(y), g(y), g(g(g(y))))$

$\sigma = \{x \rightarrow g(y)\}$

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(g(a), g(a), g(z))$

$f(g(a), g(a), g(g(g(a))))$

$\sigma = \{x \rightarrow g(a), y \rightarrow a\}$

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(g(a), g(a), g(z))$

$f(g(a), g(a), g(g(g(a))))$

$\sigma = \{x \rightarrow g(a), y \rightarrow a\}$

Example:  $f(x, g(a), g(z)) \stackrel{?}{=} f(g(y), g(y), g(g(x)))$

$f(g(a), g(a), g(g(g(a))))$

$f(g(a), g(a), g(g(g(a))))$

$\sigma = \{x \rightarrow g(a), y \rightarrow a, z \rightarrow g(g(a))\}$

# Questions

- 1 Correctness:
  - Does the algorithm always terminate?
  - Does it always produce an mgu for two unifiable terms, and fail for non-unifiable terms?
  - Do these answers depend on the order of operations?
- 2 Complexity:
  - How much space does this take, and how much time?
- 3 Extension with equational theory, e.g.,  $ab = ba$ .

# Syntactic Unification is Unitary

## Theorem (Robinson)

*Without equational theory there exists a unique mgu for syntactic unification (modulo renaming). Unification is called unitary.*

Proof: Exercise

Herbrand, Martelli, Montanari, Plotkin, Robinson, Huet, Knuth, Bendix, Siekman, Baader.

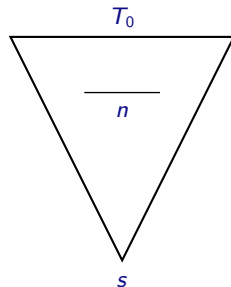
# Outline

- 1 Dolev Yao's Intruder
- 2 Terms and Messages
- 3 Notion of Locality**
- 4 Passive Intruder: Intruder Deduction Problem
- 5 Undecidability for unbounded number of sessions
- 6 Conclusion

## Definition of S-Locality

- A proof  $P$  of  $T_0 \vdash s$  is S-local :

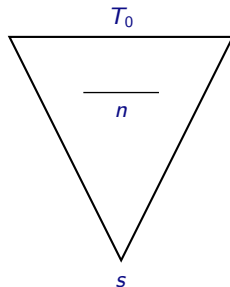
$$\forall n \in P, n \in S(T_0 \cup \{s\})$$



## Definition of S-Locality

- A proof  $P$  of  $T_0 \vdash s$  is S-local :

$$\forall n \in P, n \in S(T_0 \cup \{s\})$$



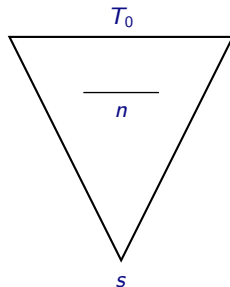
S-Local Proof:

A proof  $P$  of  $T \vdash w$  is **S-local** if all nodes are in  $S(T \cup \{w\})$ .

## Definition of S-Locality

- A proof  $P$  of  $T_0 \vdash s$  is S-local :

$$\forall n \in P, n \in S(T_0 \cup \{s\})$$



### S-Local Proof:

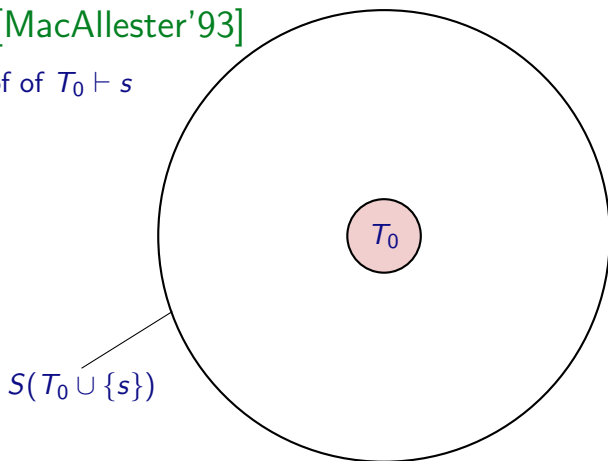
A proof  $P$  of  $T \vdash w$  is **S-local** if all nodes are in  $S(T \cup \{w\})$ .

### S-Locality :

A proof system is **S-local** if whenever there is a proof of  $T \vdash w$  then there is also a S-local proof of  $T \vdash w$ .

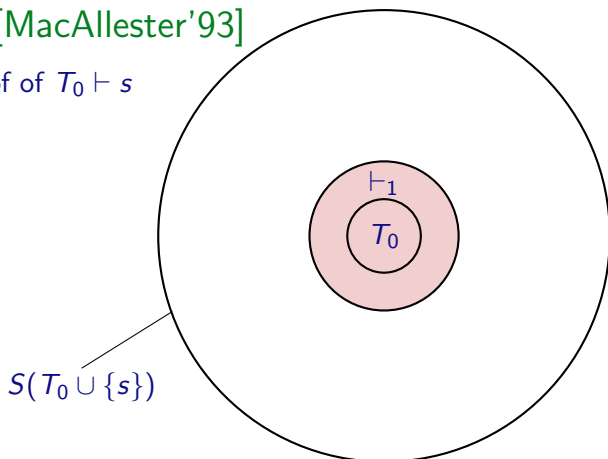
## Locality Idea [MacAllester'93]

P a  $S$ -local proof of  $T_0 \vdash s$



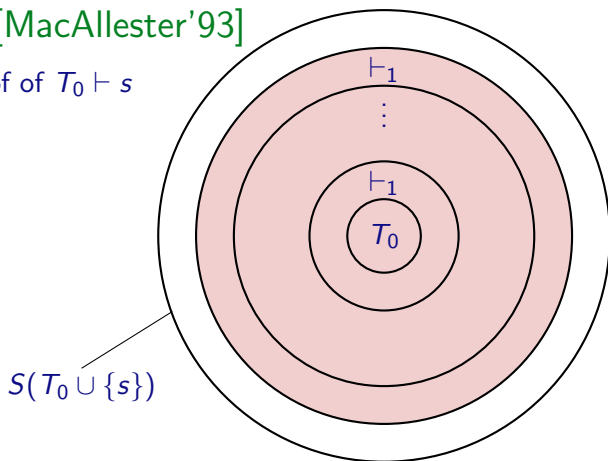
## Locality Idea [MacAllester'93]

P a S-local proof of  $T_0 \vdash s$



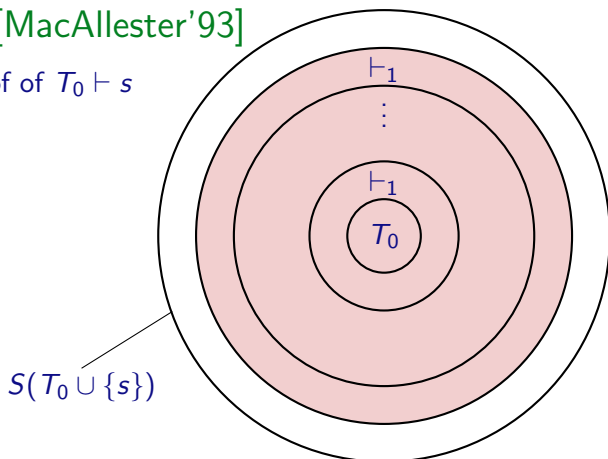
## Locality Idea [MacAllester'93]

P a S-local proof of  $T_0 \vdash s$



## Locality Idea [MacAllester'93]

P a S-local proof of  $T_0 \vdash s$



Intruder Deduction Problem :  $T_0 \vdash^? s$

- S-locality
- One-step deductibility

Example:  $T_0 \vdash s$  is it a local proof?

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = b$

$$\begin{array}{c}
 \begin{array}{c}
 (A) \frac{\langle a, \{c\}_k \rangle \in T_0}{T_0 \vdash \langle a, \{c\}_k \rangle} \\
 (UR) \frac{}{T_0 \vdash \{c\}_k} \\
 (D) \frac{}{T_0 \vdash c}
 \end{array}
 \quad
 \begin{array}{c}
 (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array}
 \quad
 \begin{array}{c}
 (A) \frac{\{b\}_c \in T_0}{T_0 \vdash \{b\}_c}
 \end{array} \\
 \hline
 (D) \frac{}{T_0 \vdash b}
 \end{array}$$

Example:  $T_0 \vdash s$  is it a local proof?

Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = b$

$$\begin{array}{c}
 \text{(A)} \frac{\langle a, \{c\}_k \rangle \in T_0}{T_0 \vdash \langle a, \{c\}_k \rangle} \\
 \text{(UR)} \frac{}{T_0 \vdash \{c\}_k} \\
 \text{(D)} \frac{}{T_0 \vdash c} \\
 \text{(A)} \frac{k \in T_0}{T_0 \vdash k} \\
 \text{(A)} \frac{\{b\}_c \in T_0}{T_0 \vdash \{b\}_c} \\
 \text{(D)} \frac{}{T_0 \vdash b}
 \end{array}$$

$$S(T_0 \cup \{s\}) = T_0 \cup \{a, b, c, \{c\}_k\}$$

## Locality Theorem

### Theorem of Locality [McAllester 93]

If a proof system  $P$  is SyntacticSubterm-local then there is a  $P$ -time procedure to decide the deducibility in  $P$ .

## Locality Theorem

### Theorem of Locality [McAllester 93]

If a proof system  $P$  is SyntacticSubterm-local then there is a  $P$ -time procedure to decide the deducibility in  $P$ .

### Restrictions:

- Deduction system must be finite
- Use just syntactic subterms

Exercise: Proof of McAllester theorem.

## Adapted McAllester Results

### McAllester's Algorithm

Input :  $T_0, w$

$T \leftarrow T_0;$

while  $(\exists s \in S(T_0, w)$  such that  $T \vdash^{\leq 1} s$  and  $s \notin T)$

$T \leftarrow T \cup \{s\};$

Output :  $w \in T$

### Theorem

Let be  $P$  a proof system, if:

- the size of  $S(T)$  is polynomial in the size of  $T$ ,
- $P$  is S-local,
- one-step deducibility is P-time decidable,

then provability in the proof system  $P$  is P-time decidable.

# Outline

- 1 Dolev Yao's Intruder
- 2 Terms and Messages
- 3 Notion of Locality
- 4 Passive Intruder: Intruder Deduction Problem**
- 5 Undecidability for unbounded number of sessions
- 6 Conclusion

## Locality Theorem

### Theorem of Locality [McAllester 93]

If a proof system  $P$  is SyntacticSubterm-local then there is a  $P$ -time procedure to decide the deducibility in  $P$ .

## Locality Theorem

### Theorem of Locality [McAllester 93]

If a proof system  $P$  is SyntacticSubterm-local then there is a  $P$ -time procedure to decide the deducibility in  $P$ .

Exercise:

Prove that Dolev Yao deduction system is S-local.

## Example I

GOAL: Find a good  $S$ !

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = b$

## Example I

GOAL: Find a good  $S$ !

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = b$

$$\begin{array}{c}
 \begin{array}{c}
 (A) \frac{\langle a, \{c\}_k \rangle \in T_0}{T_0 \vdash \langle a, \{c\}_k \rangle} \\
 (UR) \frac{}{T_0 \vdash \{c\}_k} \\
 (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array} \\
 \hline
 \begin{array}{c}
 (A) \frac{\{b\}_c \in T_0}{T_0 \vdash \{b\}_c} \\
 (D) \frac{}{T_0 \vdash c}
 \end{array} \\
 \hline
 (D) \frac{}{T_0 \vdash b}
 \end{array}$$

## Example I

GOAL: Find a good  $S$ !

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = b$

$$\begin{array}{c}
 \begin{array}{c}
 (A) \frac{\langle a, \{c\}_k \rangle \in T_0}{T_0 \vdash \langle a, \{c\}_k \rangle} \\
 (UR) \frac{}{T_0 \vdash \{c\}_k} \\
 (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array} \\
 \hline
 \begin{array}{c}
 (A) \frac{\{b\}_c \in T_0}{T_0 \vdash \{b\}_c} \\
 (D) \frac{}{T_0 \vdash c}
 \end{array} \\
 \hline
 (D) \frac{}{T_0 \vdash b}
 \end{array}$$

$S(T_0) = T_0 \cup \{s, a, b, c, k, \{b\}_c, \{c\}_k\}$

## Example II

GOAL: Find a good  $S$ .

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = \langle b, k \rangle$

## Example II

GOAL: Find a good  $S$ .

### Example

$T_0 = \{k, \{b\}_c, \langle a, \{c\}_k \rangle\}$  and  $s = \langle b, k \rangle$

$$\begin{array}{c}
 \begin{array}{c}
 (A) \frac{\langle a, \{c\}_k \rangle \in T_0}{T_0 \vdash \langle a, \{c\}_k \rangle} \\
 (UR) \frac{}{T_0 \vdash \{c\}_k} \\
 (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array} \\
 \hline
 (D) \frac{(A) \frac{\{b\}_c \in T_0}{T_0 \vdash \{b\}_c} \quad (D) \frac{}{T_0 \vdash c}}{T_0 \vdash b} \\
 \hline
 (P) \frac{}{T_0 \vdash \langle b, k \rangle} \quad (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array}$$



## Example III

GOAL: Find a good  $S$ .

### Example

$T_0 = \{k, \{c\}_k\}$  and  $s = c$

## Example III

GOAL: Find a good  $S$ .

### Example

$T_0 = \{k, \{c\}_k\}$  and  $s = c$

$$\begin{array}{c}
 \frac{(A) \frac{\{c\}_k \in T_0}{T_0 \vdash \{c\}_k} (A) \frac{\{c\}_k \in T_0}{T_0 \vdash \{c\}_k}}{(P) \frac{}{T_0 \vdash \langle \{c\}_k, \{c\}_k \rangle}} \\
 \frac{(UL) \frac{}{T_0 \vdash \{c\}_k}}{(D) \frac{}{c}} \quad (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array}$$

## Example III

GOAL: Find a good  $S$ .

### Example

$T_0 = \{k, \{c\}_k\}$  and  $s = c$

$$\begin{array}{c}
 \frac{(A) \frac{\{c\}_k \in T_0}{T_0 \vdash \{c\}_k} (A) \frac{\{c\}_k \in T_0}{T_0 \vdash \{c\}_k}}{(P) \frac{}{T_0 \vdash \langle \{c\}_k, \{c\}_k \rangle}} \\
 \frac{(UL) \frac{}{T_0 \vdash \{c\}_k}}{(D) \frac{}{c}} \quad (A) \frac{k \in T_0}{T_0 \vdash k}
 \end{array}$$

$S(T_0) = T_0 \cup \{c\}$  but  $\langle \{c\}_k, \{c\}_k \rangle$

It is Not in  $S(T_0)$

## Simple Proof

- A **simple proof** is a proof where each node  $T \vdash v$  occurs at most once on each branch.
- A proof  $P$  of  $T \vdash u$  is **minimal** if there is no proof  $P'$  of  $T \vdash u$  such that  $|P'| < |P|$ .

$P$  Minimal  $\Rightarrow P$  Simple

If  $P$  is a minimal proof of  $T \vdash u$  then  $P$  is a simple proof of  $T \vdash u$ .

Proof: Exercise.

## Proof

Let us assume to the contrary that  $P$  is a non-simple proof of  $T \vdash u$ . Then there is a branch of  $P$  in which  $T \vdash v$  occurs twice. We can cut the derivation between these two occurrences and so get a smaller proof  $P'$ , which is in contradiction to the minimality of  $P$ .

## Exercises ...

We propose the following procedure to know if  $\{t_1, \dots, t_n\} \vdash_{\mathcal{I}_{DY}} t$ :

1. Apply the rules decomposition ( $D$ ), ( $UR$ ), ( $UL$ ) first  $\frac{\{x\}_y \ y}{x}$   
and  $\frac{\langle x_1, x_2 \rangle}{x_i}$  until to reach a fix point.
2. Try to build  $t$  from a set of terms get using only composition rules ( $P$ ), ( $C$ )  $\frac{x \ y}{\{x\}_y}$  and  $\frac{x_1 \ x_2}{\langle x_1, x_2 \rangle}$ .

Why this procedure is false?

What is the restriction we have to add to get this result true?

## Counter Example

### Composed keys

$$T_0 = \{k, \{c\}_{\langle k,k \rangle}\} \text{ and } s = c$$

# Counter Example

## Composed keys

$T_0 = \{k, \{c\}_{\langle k,k \rangle}\}$  and  $s = c$

$$(D) \frac{(A) \frac{\{c\}_{\langle k,k \rangle} \in T_0}{\{c\}_{\langle k,k \rangle}} (P) \frac{(A) \frac{k \in T_0}{k} (A) \frac{k \in T_0}{k}}{\langle k,k \rangle}}{c}$$

## Counter Example

### Composed keys

$T_0 = \{k, \{c\}_{\langle k,k \rangle}\}$  and  $s = c$

$$(D) \frac{(A) \frac{\{c\}_{\langle k,k \rangle} \in T_0}{\{c\}_{\langle k,k \rangle}} (P) \frac{(A) \frac{k \in T_0}{k} (A) \frac{k \in T_0}{k}}{\langle k,k \rangle}}{c}$$

True if we avoid composed keys, i.e., only with atomic keys

## Lemma (I)

Let  $P'$  be a simple proof of the form:

$$P' = \left\{ \begin{array}{c} P'_1 \dots P'_n \\ \hline T \vdash w \end{array} \right.$$

- 1 If  $T \vdash u$  does not occur in any of  $P'_1, \dots, P'_n$  and  $\langle u, v \rangle \in S(w)$  **then** there is at least one  $P'_i$  and there exists  $w'$  such that  $\langle u, v \rangle \in S(w')$  and either the root of  $P'_i$  is  $T \vdash w'$  or  $w' \in T$ .
- 2 If  $T \vdash u$  does not occur in any of  $P'_1, \dots, P'_n$  and  $\{u\}_v \in S(w)$  **then** there is at least one  $P'_i$  and there exists  $w'$  such that  $\{u\}_v \in S(w')$  and either the root of  $P'_i$  is  $T \vdash w'$  or  $w' \in T$ .

## Proof of Lemma (I)

We only give the proof for the first case (pairing), the second case is similar. We consider all possible rules for the root of  $P'$ :

- The last rule is (A) it is obvious.
- The last rule is (UL) or (UR):  $\langle u, v \rangle \in S(w)$  by hypothesis, and by construction  $w \in S(\langle u_1, u_2 \rangle)$ . We deduce by transitivity of the subterm relation that  $\langle u, v \rangle \in S(\langle u_1, u_2 \rangle)$ .
- The last rule is (D): as in the above case but with  $\{u\}_v$  instead of  $\langle u, v \rangle$ .
- The last rule is (P): since  $T \vdash u$  can not be in  $P$  then  $w = \langle w_1, w_2 \rangle \neq \langle u, v \rangle$ . But  $\langle u, v \rangle$  is a subterm of  $w$  so it is a subterm of  $w_1$  or of  $w_2$ .
- The last rule is (C): since  $T \vdash u$  can not be in  $P$  then  $w = \{w_1\}_{w_2} \neq \{u\}_v$ . But  $\langle u, v \rangle$  is a subterm of  $w$  so it is a subterm of  $w_1$  or of  $w_2$ .

## Lemma (UL) (UR)

Let  $P$  be a simple proof of  $T \vdash u$ . If  $P$  is of the form:

$$(UL) \frac{(X) \frac{\vdots}{T \vdash \langle u, v \rangle}}{T \vdash u}$$

or

$$(UR) \frac{(X) \frac{\vdots}{T \vdash \langle u, v \rangle}}{T \vdash v}$$

Then  $\langle u, v \rangle \in S(T)$

## Proof of Lemma (UL), (UR) is similar

$$(UL) \frac{(X) \frac{P_1 \dots P_n}{T \vdash \langle u, v \rangle}}{T \vdash u}$$

Since  $P$  is simple then  $T \vdash u$  does not occur in any  $P_i$ .

Using Lemma I, either  $\langle u, v \rangle \in T$ , either there is at least one  $P_i$  and there exists  $w$  such that  $\langle u, v \rangle \in S(w)$  and either the root of  $P_i$  is  $T \vdash w$  or  $w \in T$ .

Applying many times Lemma I we conclude.

## Lemma (D)

Let  $P$  be a simple proof of  $T \vdash u$ . If  $P$  is of the form:

$$(D) \frac{(R) \frac{\vdots}{T \vdash \{u\}_k} (X) \frac{\vdots}{T \vdash k}}{T \vdash u}$$

Then  $\{u\}_k \in \mathcal{S}(T)$

## Proof of Lemma (D)

$$(D) \frac{\begin{array}{c} \vdots \\ (R) \frac{}{T \vdash \{u\}_k} \\ \vdots \end{array} \quad \begin{array}{c} \vdots \\ (X) \frac{}{T \vdash k} \\ \vdots \end{array}}{T \vdash u}$$

By induction on the structure of  $P$ :

- $R = A$  Trivial
- $R = UL$  or  $UL$  using Lemma  $(UR)$   $(UL)$
- $R = P$  impossible
- $R = C$  key of  $(C)$  cannot be  $k$  because  $P$  is simple. Neither  $k'$  since we obtain  $\{u\}_k$
- $R = D$  by induction hypothesis  $\{\{u\}_k\}_{k'} \in S(T)$ , then  $\{u\}_k \in S(T)$ .

## Lemma (P)

Let  $P$  be a simple proof of  $T \vdash w$ . If  $P$  contains one application of the rule  $P$

$$(P) \frac{(X) \frac{\vdots}{T \vdash u} (X) \frac{\vdots}{T \vdash v}}{T \vdash \langle u, v \rangle}$$

Then  $\langle u, v \rangle \in S(T, w)$

## Proof of Lemma (P) I

By induction on the structure of  $P$  last rule is:

- $A$ : Trivial
- $UL$  or  $UR$ : by induction and using Lemma ( $UR$ ) ( $UL$ ) then  $\langle w, v \rangle \in S(T)$  by consequence all node build from ( $P$ ) rule are in  $S(T) \subseteq S(T, w)$
- $C$ : by induction and  $u, k \in S(\{u\}_k) = S(w)$
- $P$ : by induction and  $u, k \in S(\langle u, k \rangle) = S(w)$

## Proof of Lemma (P) II

Last case!

$$(D) \frac{(X) \frac{P_1}{T \vdash \{w\}_k} (X) \frac{P_2}{T \vdash k}}{T \vdash w}$$

- *D*: Consider occurrences of (*P*) generating  $v = \langle v_1, v_2 \rangle$  in  $P_1$  and  $P_2$ . By induction  $v \in S(T, k)$  and  $v \in S(T, \{w\}_k)$ . Assume  $v \notin S(T, w)$  implies that  $v \in S(T, k)$  (because  $v$  is a pair). Otherwise occurrences of  $v$  in  $k$  are canceled by (*D*)  
 If keys are atomics, we know that  $k$  comes from  $S(T)$  using Lemma (UL) (UR) or (D). (“wrong” lemma is true in this case)

## Proof of Lemma (P) III

We analyze path from  $\{w\}_k$  and  $k$  the path where we have  $v$ . If it ends on (A) then  $v \in S(T)$ , which contradicts fact that  $v \notin S(T, w)$ .

Otherwise the two paths end with application of rule (P) and we can find a new minimal proof by “cutting” the rule (P).

## Lemma (C)

Let  $P$  be a simple proof of  $T \vdash w$ . If  $P$  contains one application of the rule  $C$

$$(P) \frac{(X) \frac{\vdots}{T \vdash u} (X) \frac{\vdots}{T \vdash v}}{T \vdash \{u\}_v}$$

Then  $\{u\}_v \in S(T, w)$

Exercise: Proof of Lemma (C), similar than Lemma (P)

## S-Locality for Dolev Yao

Using all these lemmas we can conclude.

## Exercises ...

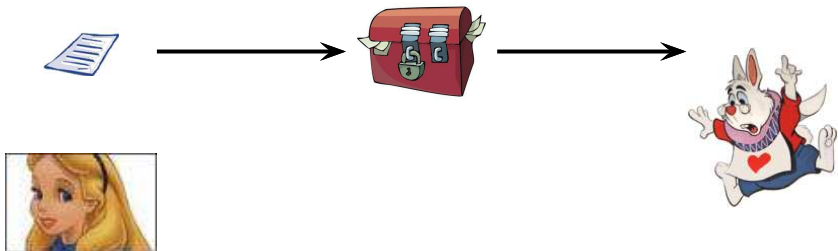
- Consider the following protocol:

$$\begin{aligned}A &\rightarrow B : \langle \{k_1\}_{k_2}, m \rangle \\ B &\rightarrow A : \{m\}_{\langle k_1, k_2 \rangle}\end{aligned}$$

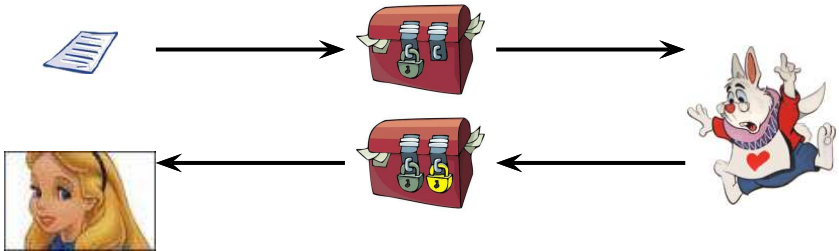
Assume that  $k_2$  is a shared key between  $A$  and  $B$ . Show that  $k_1$  is secret in presence of passive Dolev-Yao intruder.

- Give an exemple of inference system for which the locality property is false.

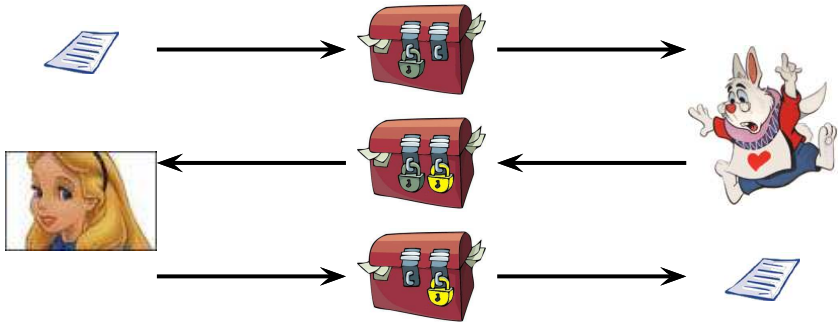
## Example :



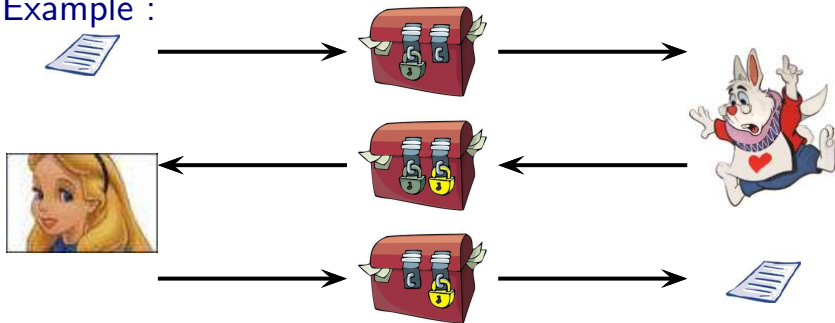
## Example :



## Example :



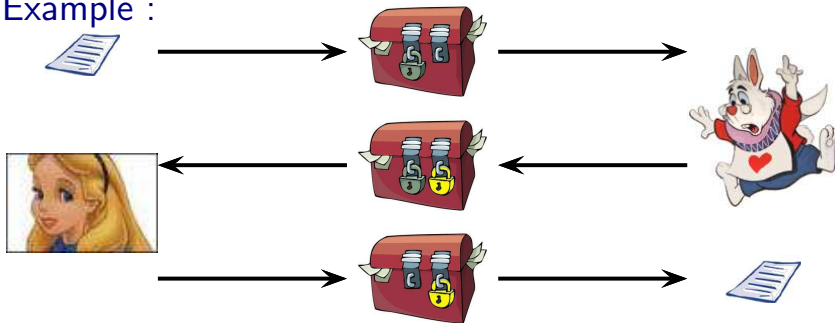
Example :



### Shamir 3-Pass Protocol

$$1 \quad A \rightarrow B : \{m\}_{K_A}$$

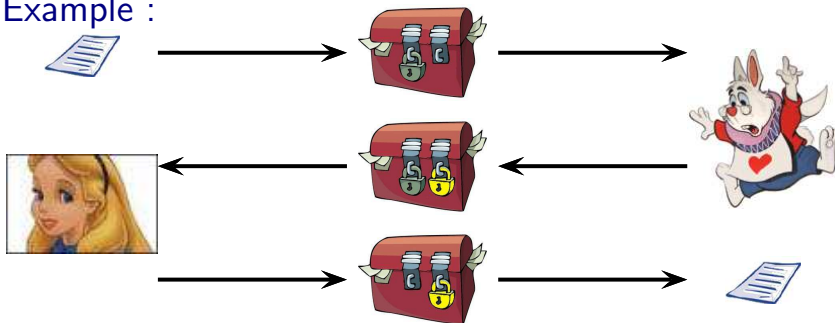
Example :



### Shamir 3-Pass Protocol

- 1  $A \rightarrow B : \{m\}_{K_A}$
- 2  $B \rightarrow A : \{\{\{m\}_{K_A}\}_{K_B}\}_{K_A}$

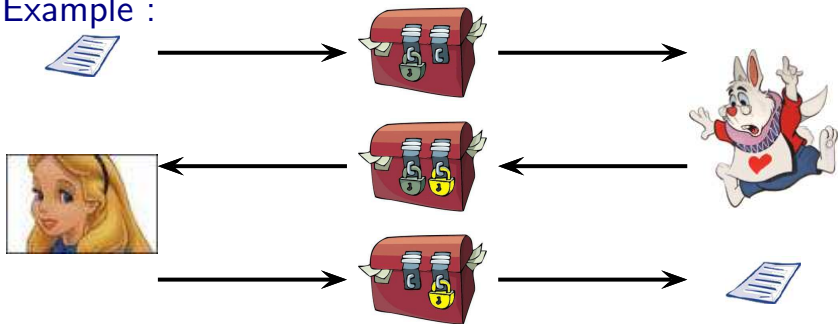
Example :



### Shamir 3-Pass Protocol

- 1  $A \rightarrow B : \{m\}_{K_A}$
  - 2  $B \rightarrow A : \{\{m\}_{K_A}\}_{K_B} = \{\{m\}_{K_B}\}_{K_A}$
- Commutative Encryption

Example :



### Shamir 3-Pass Protocol

- 1  $A \rightarrow B : \{m\}_{K_A}$
- 2  $B \rightarrow A : \{\{m\}_{K_A}\}_{K_B} = \{\{m\}_{K_B}\}_{K_A}$
- 3  $A \rightarrow B : \{m\}_{K_B}$

Commutative  
Encryption

## Logical Attack on Shamir 3-Pass Protocol (I)

Perfect encryption one-time pad (Vernam Encryption)

$$\{m\}_k = m \oplus k$$

XOR Properties (ACUN)

- $(x \oplus y) \oplus z = x \oplus (y \oplus z)$
- $x \oplus y = y \oplus x$
- $x \oplus 0 = x$
- $x \oplus x = 0$

**A**ssociativity

**C**ommutativity

**U**nity

**N**ilpotency

## Logical Attack on Shamir 3-Pass Protocol (I)

Perfect encryption one-time pad (Vernam Encryption)

$$\{m\}_k = m \oplus k$$

XOR Properties (ACUN)

- $(x \oplus y) \oplus z = x \oplus (y \oplus z)$

**A**ssociativity

- $x \oplus y = y \oplus x$

**C**ommutativity

- $x \oplus 0 = x$

**U**nity

- $x \oplus x = 0$

**N**ilpotency

Vernam encryption is a **commutative encryption** :

$$\{\{m\}_{K_A}\}_{K_I} = (m \oplus K_A) \oplus K_I = (m \oplus K_I) \oplus K_A = \{\{m\}_{K_I}\}_{K_A}$$

## Logical Attack on Shamir 3-Pass Protocol (II)

Exercise: Find an attack with passive intruder against Shamir 3-Pass Protocol using Vernam encryption scheme.

## Logical Attack on Shamir 3-Pass Protocol (II)

Perfect encryption one-time pad (Vernam Encryption)

$$\{m\}_k = m \oplus k$$

Shamir 3-Pass Protocol



- 1  $A \rightarrow B : m \oplus K_A$
- 2  $B \rightarrow A : (m \oplus K_A) \oplus K_B$
- 3  $A \rightarrow B : m \oplus K_B$



Passive attacker :

$$m \oplus K_A \quad m \oplus K_B \oplus K_A \quad m \oplus K_B$$



## Logical Attack on Shamir 3-Pass Protocol (II)

Perfect encryption one-time pad (Vernam Encryption)

$$\{m\}_k = m \oplus k$$

Shamir 3-Pass Protocol



- 1  $A \rightarrow B : m \oplus K_A$
- 2  $B \rightarrow A : (m \oplus K_A) \oplus K_B$
- 3  $A \rightarrow B : m \oplus K_B$



Passive attacker :

$$m \oplus K_A \oplus m \oplus K_B \oplus K_A \oplus m \oplus K_B = m$$



# Outline

- 1 Dolev Yao's Intruder
- 2 Terms and Messages
- 3 Notion of Locality
- 4 Passive Intruder: Intruder Deduction Problem
- 5 Undecidability for unbounded number of sessions**
- 6 Conclusion

## Main Results

In general security problem **undecidable** [DLMS'99, AC'01]

Bounded number of session  $\Rightarrow$  **Decidability** [AL'00, RT'01]

# Undecidability

## Definition (Post Correspondance Problem (PCP))

Let  $\Sigma$  be a finite alphabet.

**Input** : Sequence of pairs  $\langle u_i, v_i \rangle_{1 \leq i \leq n}$   $u_i, v_i \in \Sigma^*$ ,  $n \in \mathbb{N}$

**Question** : Existence of  $k, i_1, \dots, i_k \in \mathbb{N}$  such that

$$u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}?$$

# Undecidability

## Definition (Post Correspondance Problem (PCP))

Let  $\Sigma$  be a finite alphabet.

**Input** : Sequence of pairs  $\langle u_i, v_i \rangle_{1 \leq i \leq n}$   $u_i, v_i \in \Sigma^*$ ,  $n \in \mathbb{N}$

**Question** : Existence of  $k, i_1, \dots, i_k \in \mathbb{N}$  such that

$$u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}?$$

## Example

$u_1$	$u_2$	$u_3$	$u_4$	$v_1$	$v_2$	$v_3$	$v_4$
<i>aba</i>	<i>bbb</i>	<i>aab</i>	<i>bb</i>	<i>a</i>	<i>aaa</i>	<i>abab</i>	<i>babba</i>

Solution: **1431**

$$u_1 \cdot u_4 \cdot u_3 \cdot u_1 = aba \cdot bb \cdot aab \cdot aba = a \cdot babba \cdot abab \cdot a = v_1 \cdot v_4 \cdot v_3 \cdot v_1$$

But no solution for  $\langle \mathbf{u}_1, \mathbf{v}_1 \rangle, \langle \mathbf{u}_2, \mathbf{v}_2 \rangle, \langle \mathbf{u}_3, \mathbf{v}_3 \rangle$

# Undecidability

## Definition (Post Correspondance Problem (PCP))

Let  $\Sigma$  be a finite alphabet.

**Input** : Sequence of pairs  $\langle u_i, v_i \rangle_{1 \leq i \leq n}$   $u_i, v_i \in \Sigma^*$ ,  $n \in \mathbb{N}$

**Question** : Existence of  $k, i_1, \dots, i_k \in \mathbb{N}$  such that

$$u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k}?$$

## Example

$u_1$	$u_2$	$u_3$	$u_4$	$v_1$	$v_2$	$v_3$	$v_4$
<i>aba</i>	<i>bbb</i>	<i>aab</i>	<i>bb</i>	<i>a</i>	<i>aaa</i>	<i>abab</i>	<i>babba</i>

Solution: **1431**

$$u_1 \cdot u_4 \cdot u_3 \cdot u_1 = aba \cdot bb \cdot aab \cdot aba = a \cdot babba \cdot abab \cdot a = v_1 \cdot v_4 \cdot v_3 \cdot v_1$$

But no solution for  $\langle \mathbf{u}_1, \mathbf{v}_1 \rangle, \langle \mathbf{u}_2, \mathbf{v}_2 \rangle, \langle \mathbf{u}_3, \mathbf{v}_3 \rangle$

PCP is undecidable

## Undecidability for Protocols

We construct a protocol such that decidability of secret implies decidability of PCP.

$$A : \text{ send}(\{\langle u_i, v_i \rangle\}_{K_{ab}}) \quad (1 \leq i \leq n)$$

$$B : \text{ receive}(\{\langle x, y \rangle\}_{K_{ab}}) \\ \text{ send}(\{\langle \langle x \cdot u_i, y \cdot v_i \rangle\}_{K_{ab}}, \{s\}_{\{\langle \langle x \cdot u_i, x \cdot u_i \rangle\}_{K_{ab}}}\}) \quad (1 \leq i \leq n)$$

We assume that  $\mathbf{K}_{AB}$  is a shared key between **A** and **B**.

Intruder can find  $\mathbf{s}$  iff he can solve PCP.

# Outline

- 1 Dolev Yao's Intruder
- 2 Terms and Messages
- 3 Notion of Locality
- 4 Passive Intruder: Intruder Deduction Problem
- 5 Undecidability for unbounded number of sessions
- 6 Conclusion**

# Summary

## Today

- Dolev Yao Model
- Terms and Messages
- Notion of Locality
- Undecidability Result

## Next Time

- Active Intruder
- Constraints
- Resolution
- Pi Calculus
- BAN Logic
- Principles using Examples
- Playing with Tools:
  - Scyther
  - Avispa: OFMC, CI-Atse, SATMC, TA4SP
  - Proverif

Thank you for your attention



Questions ?