Formalization of Security Aspects, Protocol Analysis and Verification

Ph.D. Thesis Proposal
Pisa, 29 January 2001

E. Tuosto

Dipartimento Di Informatica, Università di Pisa

Outline

- Security: what is it?
- "Simply" . . . properties of systems
- Analysis and Verification
- Attack example
- Formal methods for security
  - Spi
  - Strand spaces
- Some proposals

Introduction

Security

Analysis and Verification

properties

(not formally specified)

systems

(formal)

Many hypothesis are not explicit

$\boxed{\text{Protocol Specification}}$

Many proposals

- Informal languages
  $(i) \quad A_i \rightarrow B_i : \quad m_i$

- Formal
  spi, logics, strands...

$$\Downarrow$$

Some aspects are not explicitly captured

(spi does not have an explicit notion principal)

$\boxed{\text{A question}}$

Are proposed languages "enough"?

$\boxed{\text{Verification}}$

- Protocols are "short", but...
- their properties must hold for all "hostile" environments

$$\Downarrow$$

state explosion

Protocol Specification

Informal notation: $(i) \quad A_i \rightarrow B_i : \quad m_i$
the protocol *designer* intended that the principal A_i sends m_i as the i-th message in the protocol, and m_i must be received by B_i

Goal: define a protocol that respects a given property

Something lacks:

1. initial knowledge (and its evolution) of principals
2. semantics of message tests
3. are protocol steps ordered?
4. what about multi-sessions?

Security Properties (1)

Are stated in terms of:

- protocol runs
- terminating protocol runs
- protocol sessions
- uniqueness of nonces and session keys
- . . .

Security Properties (2)

- Secrecy (strictly related to information flow)
- Integrity
- Authentication (many versions: classified by Lowe)
- but also Denial of service, fairness, non-repudiation

Protocol Analysis (1)

Hypothesis

- Perfect encryption
- Dolev-Yao attacker

$\boxed{\text{L vs. N/S: 1-0}}$

Needham-Schroeder Authentication Protocol:

$$
\begin{array}{ll}
(1) & A \rightarrow B: \ \{N_a\ A\}_{k_B} \\
(2) & B \rightarrow A: \ \{N_a\ N_b\}_{k_A} \\
(3) & A \rightarrow B: \ \{N_b\}_{k_B}
\end{array}
$$

The attack of Lowe:

$$
\begin{array}{lll}
(1) & A \rightarrow I: & \{N_a\ A\}_{k_I} \\
(1') & I \rightarrow B: & \{N_a\ A\}_{k_B} \\
(2 = 2') & B \rightarrow A: & \{N_a\ N_b\}_{k_A} \\
(3) & A \rightarrow I: & \{N_b\}_{k_I} \\
(3') & I \rightarrow B: & \{N_b\}_{k_B}
\end{array}
$$

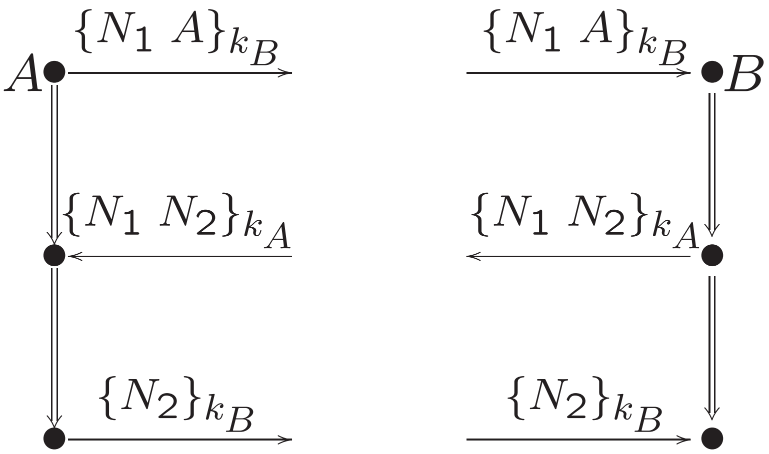
Formal Methods and Security

Many proposals

- Process Algebras (SPA, CSP, spi, spi−−...)
- Static analysis
- Types
- Logic

Focus on strand spaces & spi

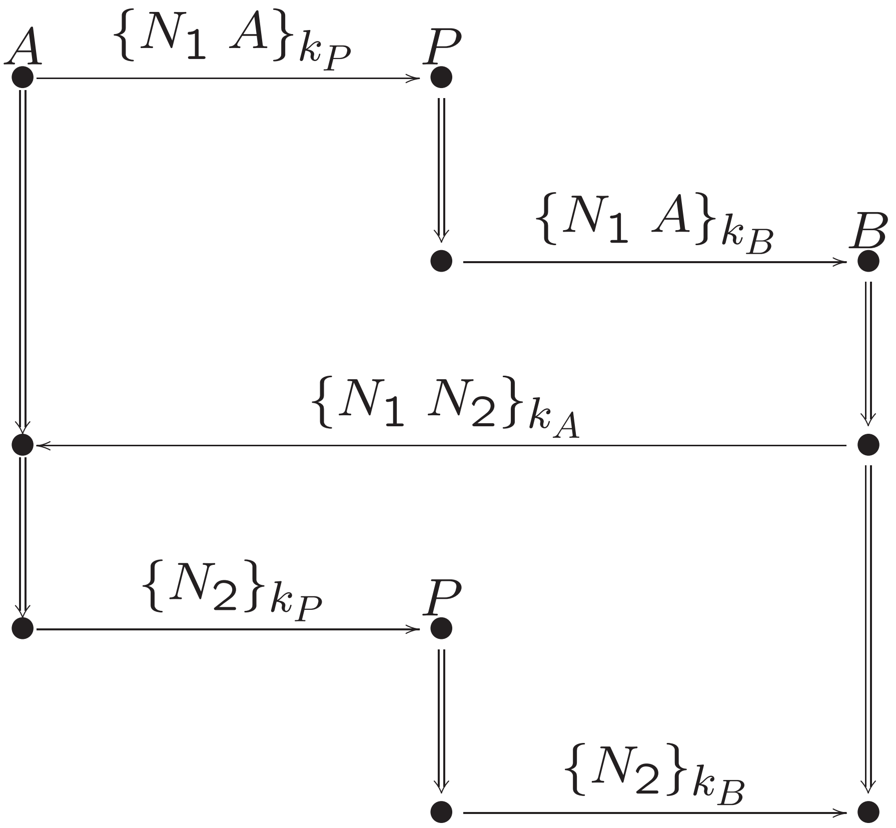
Strand Spaces (1)



Penetrator

`M.`	*Text message:*	$\langle +t \rangle$	$t \in \mathbf{T}$
`F.`	*Flushing:*	$\langle -g \rangle$	
`T.`	*Tee:*	$\langle -g, +g, +g \rangle$	
`C.`	*Concatenation:*	$\langle -g, -h, +gh \rangle$	
`S.`	*Splitting:*	$\langle -gh, +g, +h \rangle$	
`K.`	*Key:*	$\langle +k \rangle$	$k \in \mathbf{K}$
`E.`	*Encryption:*	$\langle -k, -h, \{h\}_k \rangle$	
`D.`	*Decryption:*	$\langle -k^{-1}, \{h\}_k, +h \rangle$	

Strand Spaces (2)



$\boxed{\text{Spi calculus (1)}}$

M, N	$::=$	**terms**	
	n	channels	
	k	keys	
	(M, N)	pair	
	$\{M\}_k$	encryption	

act	$::=$	**actions**	
	$\overline{N}\langle M \rangle$		
	$N(x)$		
p, q, r	$::=$		**processes**
	$act.p$		prefix
	$p \mid q$		composition
	$(\nu\, n)p$		restriction
	$!p$		bang
	$[M = N]p$		match
	0		nil
	let $(x, y) = M$ **in** p		pair splitting
	case M **of** $\{x\}_k$ **in** p		decryption

Spi Calculus (2)

Attacker is not *explicitly* modeled!

Properties are stated in terms of behavioural equations (testing equivalence)

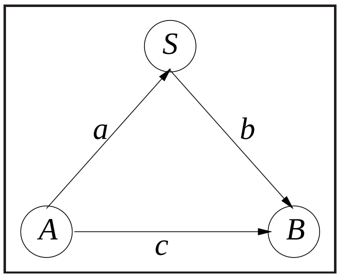
For instance, a protocol $P(x)$ respects the secrecy of its parameter x if

$$\forall m, m'. F[m] \simeq F[m'] \Longrightarrow P(m) \simeq P(m')$$

(F[x] is the continuation of the protocol)

Spi Calculus (3)

Wide Mouthed Frog:



$$\begin{array}{llll} (1) & A \longrightarrow S: & \{k\}_{k_{as}} & \text{on } a \\ (2) & S \longrightarrow B: & \{k\}_{k_{bs}} & \text{on } b \\ (3) & A \longrightarrow B: & \{M\}_{k} & \text{on } c \end{array}$$

$$\begin{array}{rcl} A(M) & \triangleq & (\nu\, k)(\overline{a}\langle\{k\}_{k_{as}}\rangle.\overline{c}\langle\{M\}_k\rangle) \\ S & \triangleq & a(x).\textbf{case } x \textbf{ of } \{y\}_{k_{as}} \textbf{ in } \overline{b}\langle\{y\}_{k_{bs}}\rangle \\ B & \triangleq & b(x).\textbf{case } x \textbf{ of } \{y\}_{k_{bs}} \textbf{ in } c(z). \\ & & \textbf{case } z \textbf{ of } \{w\}_y \textbf{ in } F[w] \\ \textit{wmf}(M) & \triangleq & (\nu\, k_{as}, k_{bs})(A(M) \mid S \mid B) \end{array}$$

wmf(M) respects the *secrecy* of a M, if

$$\forall m, m'.F[m] \simeq F[m'] \Longrightarrow \textit{wmf}(m) \simeq \textit{wmf}(m')$$

$\boxed{\text{Proposals}}$

$\boxed{\text{Specification (1)}}$

Goals:

1. Give a formalism whose constructs are related to security properties

   linguistic mechanism adequate to “what” properties speaks about

2. new mechanisms for expressing new “kind” of protocols and properties

   Security as Quality of Service

. . . by extending spi

Specification (2)

Goal 1
spi $+$ features for explicitly expressing some concepts that occur in property specifications.

- names for protocols ($\mathcal{P}$);
- names for principals ($\mathcal{A}$);
- names for nonces ($\mathcal{N}$);

A protocol:
$!p(\tilde{A}, \tilde{k}, \tilde{N}, \tilde{m}).P$

Contexts:
$C[\bullet] \quad ::= \quad \bullet, \ \ P, \ \ \overline{a}(M).C, \ \ C \mid C', \ \ C + C'$

Property scheme:
$\forall tr \in \mathbf{Tr}(P) : tr \downarrow_{\alpha} \Longrightarrow tr \downarrow_{\beta} \wedge \Re(\alpha, \beta)$

Specification (3)

Goal 2.1

- Security as a set of "system parameters"
- Different component of a system may "bargain" the value of such parameters
- Security becomes part of the programmable coordination politics of distributed systems
- QoS as a metaphor of security

$\boxed{\text{QoS + Constraints = Security}}$

- **tell**$[\![\mathcal{K}]\!]$

- **ask**$[\![\mathcal{K}]\!] \mid\Rightarrow P \diamondsuit Q$

- $\mathcal{K}$: Local constraints on keys, nonces,...

- **ask** is a refinement of match

$$\Downarrow$$

Relation between coordination and security

QoS: An example

$$
\begin{array}{rcl}
A(\mathsf{m}, M) & \triangleq & (\nu\, k)(\textbf{tell}[\![\#k = \mathsf{m}]\!].\overline{a}\langle\{k\}_{k_{as}}\rangle. \\
& & \quad (ok().\overline{c}\langle\{M\}_k\rangle + c(y).G[y])) \\
B(\mathsf{n}) & \triangleq & !(b(x).\textbf{case } x \textbf{ of } \{y\}_{k_{bs}} \textbf{ in} \\
& & \quad \textbf{ask}[\![\#y = \mathsf{n}]\!] \\
& & \quad |\!\Rightarrow \overline{ok}\langle\rangle c(z).\textbf{case } z \textbf{ of } \{w\}_y \textbf{ in } F[w] \\
& & \quad \diamondsuit\ \overline{c}\langle\mathsf{n}\rangle)
\end{array}
$$

$$
\mathit{Inst}(\mathsf{m}, \mathsf{n}, M) \triangleq (\nu\, k_{as}, k_{bs})(A(\mathsf{m}, M) \mid S \mid B(\mathsf{n}))
$$

$$
\boxed{\dfrac{\mathcal{K}_1 :: P_1 \xrightarrow{\overline{c}\langle M\rangle} \mathcal{K}_1' :: P_1' \quad \mathcal{K}_2 :: P_2 \xrightarrow{c(x)} \mathcal{K}_2' :: P_2' \qquad \mathcal{K}_1 \downarrow_M \wedge\ \mathcal{K}_2[M/x]}{\mathcal{K}_1 :: P_1 \mid \mathcal{K}_2 :: P_2 \xrightarrow{\tau} \mathcal{K}_1' :: P_1' \mid (\mathcal{K}_2' :: P_2')[M/x]}}
$$

Multiple Interactions

Goal 2.2

- Many applications "synchronize" more than two partners
- Often security is required
  - TV-on demand
  - Electronic conferences
  - Fairness in e-commerce applications

$\mu\pi$-calculus syntax

$$
\begin{array}{llll}
p, q & ::= & 0 & \textit{Null process} \\
& \mid & \overline{\lambda} b.0 & \textit{Multiple Output} \\
& \mid & a(b).p & \textit{Input} \\
& \mid & !p & \textit{Replication} \\
& \mid & p \mid q & \textit{Composition} \\
& \mid & (\nu a)p & \textit{Restriction} \\
& \mid & [a = b]p & \textit{Matching}
\end{array}
$$

$$
\lambda \quad ::= \quad a \mid a \wedge \lambda
$$

$$
\begin{array}{lrcll}
\text{(monoid)} & \multicolumn{4}{l}{\mid \text{ associative, commutative, } 0 \text{ identity}} \\
\text{(scope)} & p \mid (\nu a)\ q & = & (\nu a)(p \mid q) & \text{if } a \notin \mathtt{fn}(p) \\
\text{(res)} & (\nu a)\ (\nu b)\ p & = & (\nu b)\ (\nu a)\ p & \\
\text{(nil1)} & (\nu a)0 & = & 0 & \\
\text{(nil2)} & !0 & = & 0 & \\
\text{(bang)} & !p & = & p \mid !p & \\
\text{(match)} & [a = a]p & = & p &
\end{array}
$$

$\mu\pi$-calculus semantics

(in) $$a(b).p \xrightarrow{ac} p[c/b]$$

(m_out) $$\overline{\lambda}b.0 \xrightarrow{\overline{\lambda}b} 0$$

(m_sync) $$\frac{p \xrightarrow{\overline{\lambda}b} p' \quad \bigwedge_{i=1}^{n} q_i \xrightarrow{\lambda_i b} q_i'}{p \mid q \xrightarrow{\tau} p' \mid q_1 \mid \ldots \mid q_n}$$

(m_close) $$\frac{p \xrightarrow{\overline{\lambda}(b)} p' \quad \bigwedge_{i=1}^{n} q_i \xrightarrow{\lambda_i b} q_i'}{p \mid q \xrightarrow{\tau} (\nu\, b)(p' \mid q_1 \mid \ldots \mid q_n)} \quad b \notin \bigcup_{i=1}^{n} \mathtt{fn}(q_i)$$

(par) $$\frac{p \xrightarrow{\alpha} p'}{p \mid q \xrightarrow{\alpha} p' \mid q} \text{ if } \mathtt{bn}(\alpha) \cap \mathtt{fn}(q) = \emptyset$$

(res) $$\frac{p \xrightarrow{\alpha} p'}{(\nu\, a)p \xrightarrow{\alpha} (\nu\, a)p'} \text{ if } a \notin \mathtt{n}(\alpha)$$

(open) $$\frac{p \xrightarrow{\overline{\lambda}b} p'}{(\nu\, a)p \xrightarrow{\overline{\lambda}(b)} p'}$$

(cong) $$\frac{p \equiv p_1 \quad p_1 \xrightarrow{\alpha} p_2 \quad p_2 \equiv q}{p \xrightarrow{\alpha} q}$$

Strands and Tile (1)

- Strand spaces provide "graphical" reasoning...
  - but protocols must be completely specified
  - Bundles are not constructed, simply stated
- Tiles more suitably describe open systems
  - Allow geometrical reasoning
  - Have an algebraic structure (gs-graphs)
  - Security protocols are "open systems"

Strands and Tile (2)

then...

- Relations between causality and locality in security protocols may be studied

- Tiles may be exploited as coordination mechanism...then, for security

Logical framework for spi (1)

Logical frameworks: meta-languages for expressing deductive systems.

Principles

• Formulas	as	Types
• Judgments	as	Types
• Proofs	as	Terms

Tools

- Proof Editors
- Proof Checkers
- Proof Assistant

Benefits

- NO α-conversion
- NO side-conditions
- Binders are λ-Abstraction
- Communication are β-Reduction

Logical framework for spi (2)

A translation in spi has been given

But transitions have too complex types

Benefits may be obtained by simplifying

May be useful also for static analysis of spi

Verification (1)

Has to face state explosion

- Usually properties are verified on trace of the protocol in a contest
- A lot of contexts must be considered
- Intruders may deal with an infinite number of messages

$\boxed{\text{Verification (2)}}$

- HD-automata extends ordinary automata

- states and transition equipped with local names

- HD-automata symmetries has a minimal representation

$$\Downarrow$$

HD as a framework to model spi processes
Perhaps with minimal automata

Problem: spi deals with terms

$$\Downarrow$$

may be they must be extended