Control Flow Analysis for Security Protocols

Chiara Bodei and Pierpaolo Degano

Dipartimento di Informatica, Università di Pisa, Italy

Mikael Buchholtz, Han Gao, Hanne Riis Nielson, and Flemming Nielson

Informatics and Mathematical Modelling,

Technical University of Denmark

Approach

- Define a protocol (using a process calculus)
- Define a Dolev-Yao style attacker
- Track message flow for protocol and attacker using *control flow analysis*
- If messages end up in a wrong place then there may be a problem
  - Attacker can alter message flow arbitrarily
  - Focus on encryption and decryption

Making Narrations Precise

- Typically, protocols are described by narrations and a textual description *but details may be imprecise*
- We make systematic translations of the protocol narrations into a process calculus called LySa
- LySa is inspired by the Spi-calculus but processes:
  - communicate through a global network
  - match values on input and decryption
  - use symmetric key cryptography

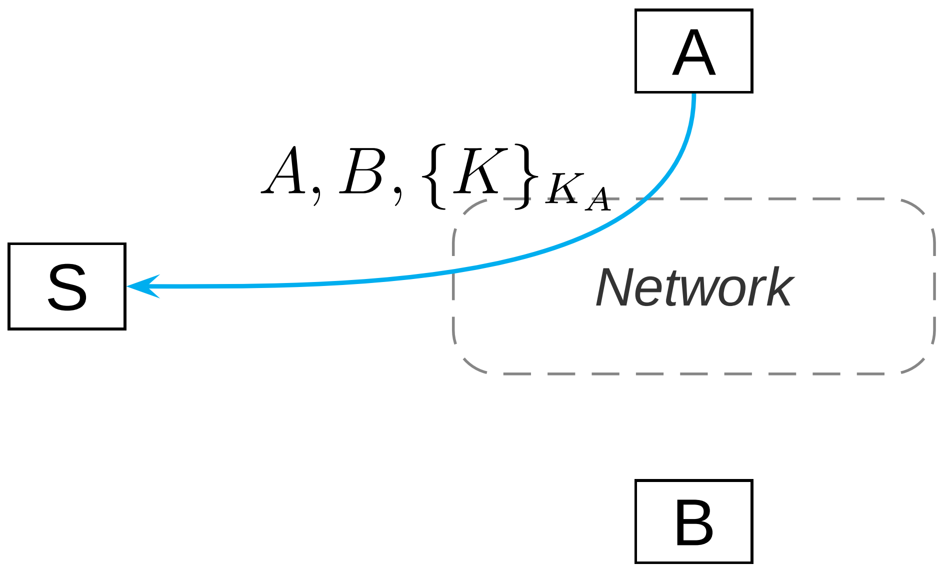
LySa Syntax

$E ::=$	n	name ($n \in \mathcal{N}$)
	x	variable ($x \in \mathcal{X}$)
	$\{E_1, \cdots, E_k\}_{E_0}$	encryption
$P ::=$	$\langle E_1, \cdots, E_k \rangle. \, P$	output
	$(E_1, \cdots, E_j; x_{j+1}, \cdots, x_k). \, P$	input (with matching)
	decrypt E as $\{E_1, \cdots, E_j; x_{j+1}, \cdots, x_k\}_{E_0}$ in P	decryption (with matching)
	$P_1 \mid P_2$	parallel composition
	$(\nu\, n)P$	introduce new name n
	$!\, P$	replication
	0	terminated process

Encoding a Protocol in LySa

A key exchange inspired protocol by the Wide Mouthed Frog:

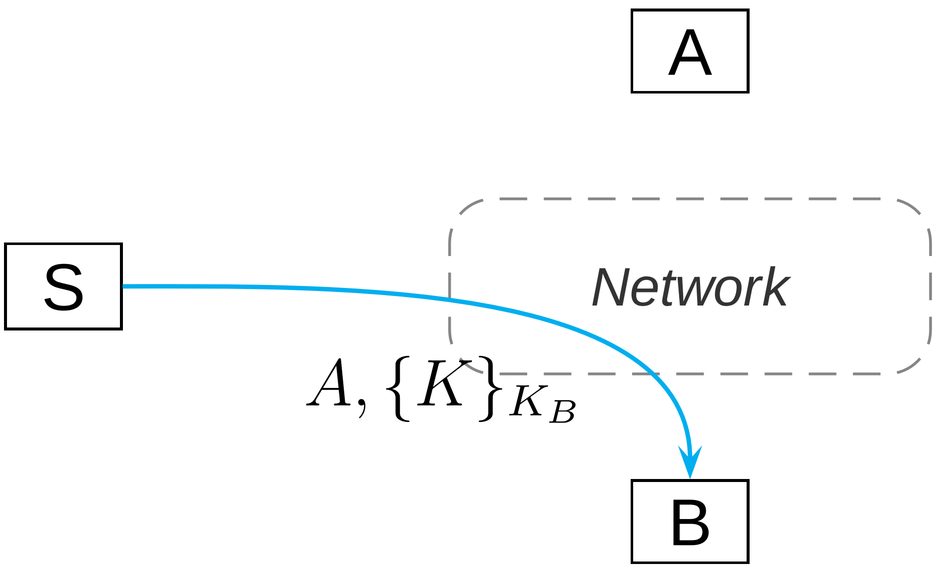
$$
\begin{array}{lll}
1. & A \rightarrow S: & A, B, \{K\}_{K_A} \\
2. & S \rightarrow B: & A, \{K\}_{K_B} \\
3. & A \rightarrow B: & \{m_1, \cdots, m_k\}_K
\end{array}
$$



Encoding a Protocol in LySa

A key exchange inspired protocol by the Wide Mouthed Frog:

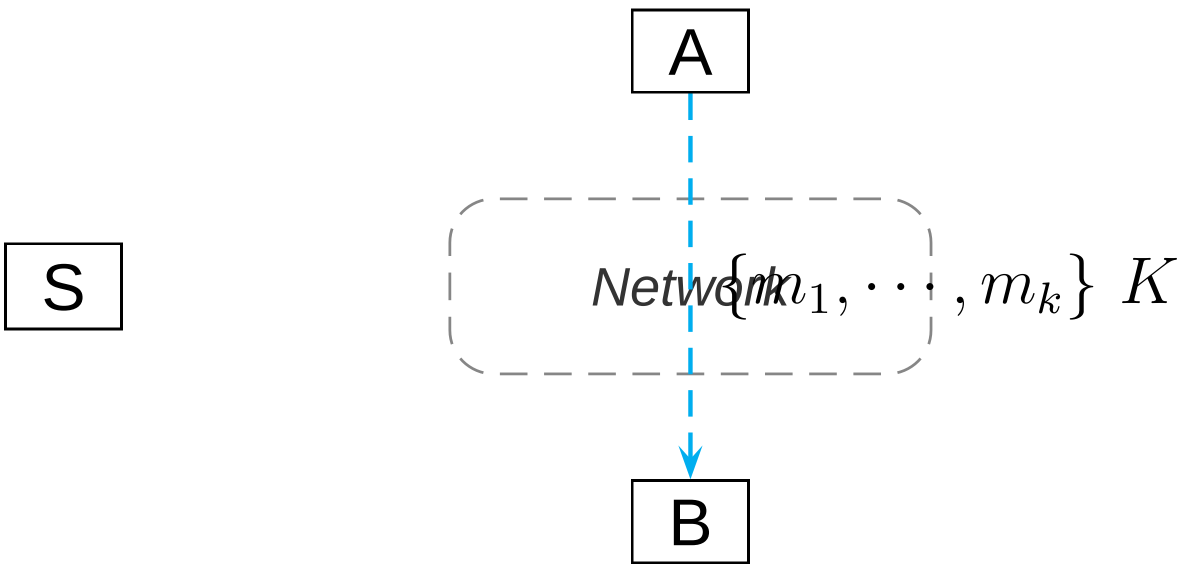
$$
\begin{array}{lll}
1. & A \rightarrow S: & A, B, \{K\}_{K_A} \\
2. & S \rightarrow B: & A, \{K\}_{K_B} \\
3. & A \rightarrow B: & \{m_1, \cdots, m_k\}_K
\end{array}
$$



Encoding a Protocol in LySa

A key exchange inspired protocol by the Wide Mouthed Frog:

1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$



Encoding a Protocol in LySa

A key exchange inspired protocol by the Wide Mouthed Frog:

1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$

$$(\nu\, K)\langle \qquad A, B, \{K\}_{K_A}\rangle.$$

Encoding a Protocol in LySa

A key exchange inspired protocol by the Wide Mouthed Frog:

1. $A \rightarrow S : \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B : \quad A, \{K\}_{K_B}$
3. $A \rightarrow B : \quad \{m_1, \cdots, m_k\}_K$

$$(\nu\, K)\langle A, S, A, B, \{K\}_{K_A}\rangle.$$

Encoding a Protocol in LySa

A key exchange inspired protocol by the Wide Mouthed Frog:

1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$

$$(\nu\, K)\langle A, S, A, B, \{K\}_{K_A}\rangle.$$

$$|$$

$$(A, S, A; x_B, x).$$

Encoding a Protocol in LySa

A key exchange inspired protocol by the Wide Mouthed Frog:

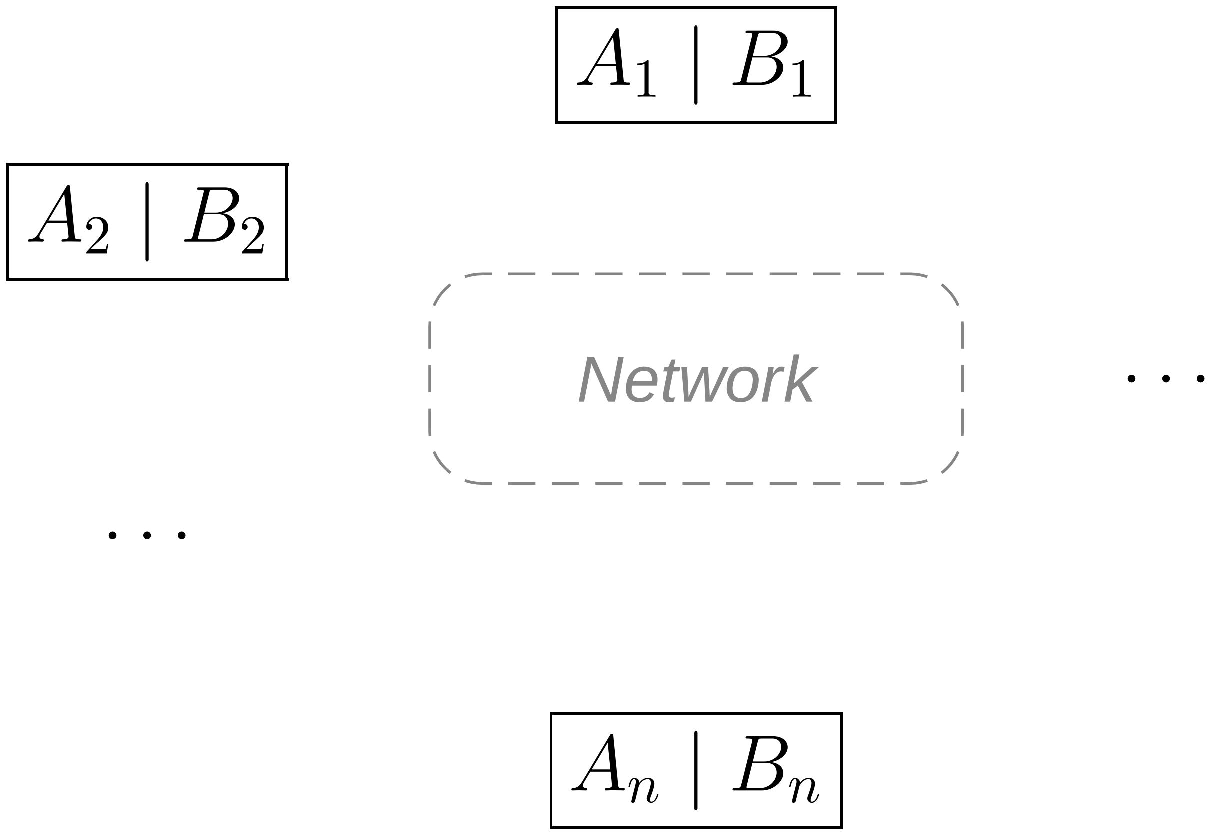
1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$

$$(\nu\, K)\langle A, S, A, B, \{K\}_{K_A}\rangle.$$

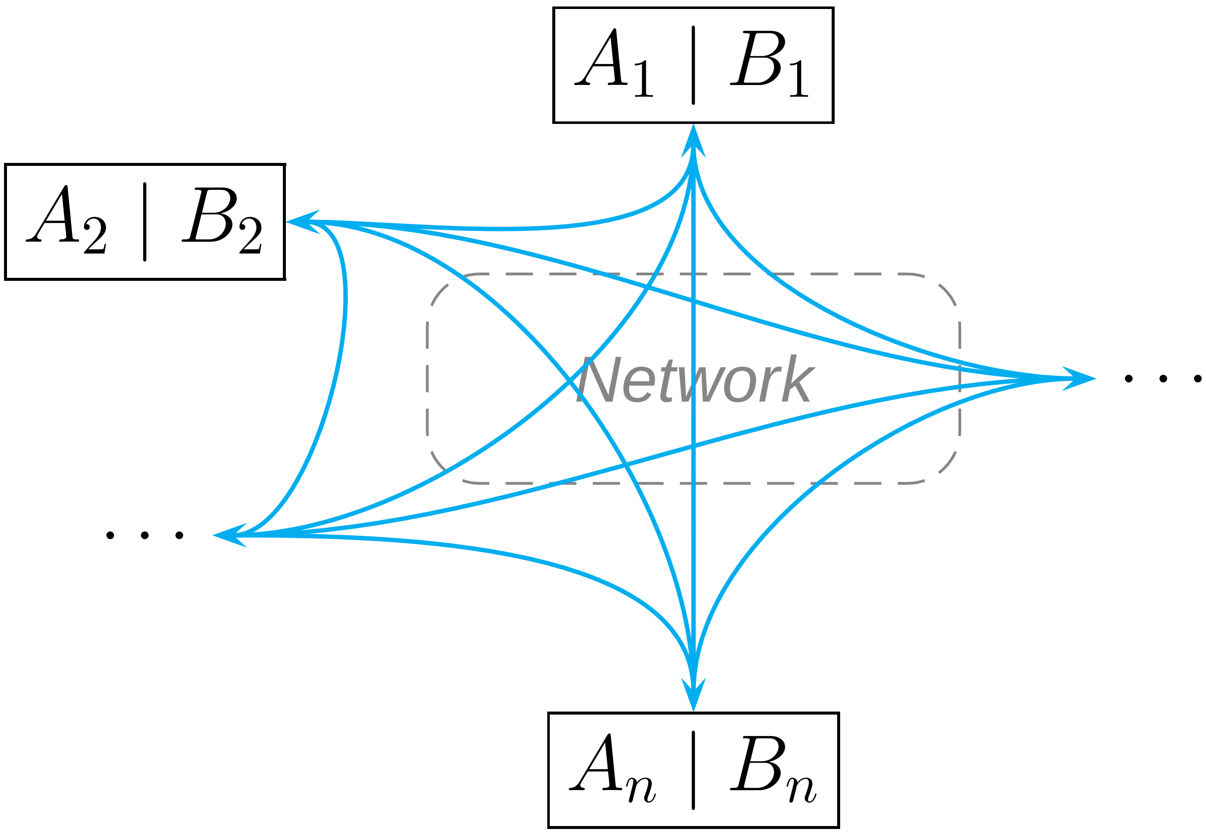
$$|$$

$(A, S, A; x_B, x)$. decrypt x as $\{; x^K\}_{K_A}$ in $\ldots$

Multiple Instances



Multiple Instances



Adding Annotation

$$
\begin{array}{lll}
1. & A \rightarrow S: & A, B, \{K\}_{K_A} \\
2. & S \rightarrow B: & A, \{K\}_{K_B} \\
3. & A \rightarrow B: & \{m_1, \cdots, m_k\}_K
\end{array}
$$

$$
\begin{array}{l}
(\nu\, K)\langle A, S, A, B, \{K\}_{K_A} \qquad\qquad \rangle. \\
| \\
(A, S, A; x_B, x).\ \text{decrypt } x \text{ as } \{; x^K\}_{K_A} \qquad\qquad \text{in } \ldots
\end{array}
$$

Adding Annotation

1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$

$$(\nu\, K)\langle A, S, A, B, \{K\}^{A}_{K_A} \qquad \rangle.$$

$$|$$

$(A, S, A; x_B, x)$. decrypt x as $\{; x^K\}^{S}_{K_A}$ in $\ldots$

Adding Annotation

1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$

$$(\nu\, K)\langle A, S, A, B, \{K\}^{A}_{K_A}[\mathsf{dest}\ S]\rangle.$$

$$|$$

$$(A, S, A; x_B, x).\,\text{decrypt } x \text{ as } \{; x^K\}^{S}_{K_A} \qquad \text{in } \ldots$$

Adding Annotation

1. $A \rightarrow S : \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B : \quad A, \{K\}_{K_B}$
3. $A \rightarrow B : \quad \{m_1, \cdots, m_k\}_K$

$$(\nu\, K)\langle A, S, A, B, \{K\}^{A}_{K_A}[\mathsf{dest}\ S]\rangle.$$

$$|$$

$$(A, S, A; x_B, x).\ \text{decrypt } x \text{ as } \{; x^K\}^{S}_{K_A}[\mathsf{orig}\ A] \text{ in } \ \ldots$$

Semantics

- Standard *reduction semantics* $P \to P'$
- The standard semantics ignores annotations

- We can also make a *reference monitor semantics* $P \to_{\mathsf{RM}} P'$
- The reference monitor gets stuck when annotations are violated

- The reference monitor *aborts* the execution of P

whenever $\quad P \to^* Q \to Q'$

but $\quad P \to^*_{\mathsf{RM}} Q \not\to_{\mathsf{RM}} Q'$

The Analysis

We make a *control flow analysis* that calculates (an over-approximation) to

- the messages on the network $\kappa \in \mathcal{P}(\mathcal{V}^*)$
- the values of the variables $\rho : \mathcal{X} \to \mathcal{P}(\mathcal{V})$

where $\mathcal{V}$ is the set of values (variable-free terms).

For example:

$$\begin{array}{ll} \langle A, S, A, B, \{K\}_{K_A}^{A}[\mathsf{dest}\ S]\rangle & \in \kappa \\ \{K\}_{K_A}^{A}[\mathsf{dest}\ S] & \in \rho(x) \\ K & \in \rho(x^K) \end{array}$$

The Error Component

The *error component* ψ collects pairs of crypto-points where the assertions in annotations may be violated. For example,

$$(A, B) \in \psi$$

reads

> "Something encrypted at A may *unexpectedly* be decrypted at B."

The Analysis

The analysis is specified as a Flow Logic with judgements

$$(\rho, \kappa) \models P : \psi$$

and auxiliary judgements for terms:

$$\rho \models E : \vartheta$$

where $\vartheta \in \mathcal{P}(\mathcal{V})$ approximates the set of values that E may evaluate to

Judgement for Decryption

(of binary terms)

$\rho \models E : \vartheta \wedge$	evaluate terms
$\rho \models E_0 : \vartheta_0 \wedge \rho \models E_1 : \vartheta_1 \wedge$	and subterms
$\forall \{V_1, V_2\}^{\ell}_{V_0} [\text{dest } \mathcal{L}] \in \vartheta :$	for all encrypted term
$\quad V_0 \text{ E } \vartheta_0 \wedge V_1 \text{ E } \vartheta_1 \Rightarrow$	if values match
$\quad\quad V_2 \in \rho(x) \wedge$	x has the value V_2
$\quad\quad \ell' \notin \mathcal{L} \vee \ell \notin \mathcal{L}' \Rightarrow (\ell, \ell') \in \psi \wedge$	check annotations
$\quad\quad (\rho, \kappa) \models P : \psi$	analyse the rest

$$\overline{(\rho, \kappa) \models \text{decrypt } E \text{ as } \{E_1; x\}^{\ell'}_{E_0} \; [\text{orig } \mathcal{L}'] \text{ in } P : \psi}$$

Correctness of the Analysis

Theorem (subject reduction)

If $(\rho, \kappa) \models P : \psi$ and $P \rightarrow Q$ then also $(\rho, \kappa) \models Q : \psi$

In fact, this holds for the standard semantics as well as the reference monitor semantics.

Theorem ($\psi = \emptyset$ means we're happy)

If $(\rho, \kappa) \models P : \emptyset$ then the reference monitor cannot abort the execution of P.

The Attacker

- Dolev-Yao style attacker that can receive, send, encrypt, decrypt, etc.
- Specified at analysis level (using ρ and κ)
- The knowledge is kept in a special variable $z_\bullet$
- For example, *receive* is written as

  $$\forall \langle V_1, \cdots, V_k \rangle \in \kappa : \wedge_{i=1}^{k} V_i \in \rho(z_\bullet)$$
- There exists a process – a “hardest attacker” – which has the capabilities of this formula
- The formula also checks annotations
- The attacker has a special crypto-point called $\ell_\bullet$

Validating Authentication

Definition

P guarantees *dynamic authentication* if $P \mid Q$ cannot abort regardless of the choice of the attacker Q.

Definition

P guarantees *static authentication* if $(\rho, \kappa) \models P : \emptyset$ and $(\rho, \kappa, \emptyset)$ satisfies the formula describing the attacker.

Theorem

If P guarantees *static authentication* then P *guarantees dynamic* authentication.

Implementation

- Transform the analysis into (an extension of) Horn clauses.
- Calculate the solution using the Succinct Solver.
- Main challenge:
  - The analysis is specified using the *infinite* universe of terms.
  - Use an encoding of terms in tree grammars where terms are represented as a *finite* number of production rules.
- Runs in *polynomial time* in the size of the process P.

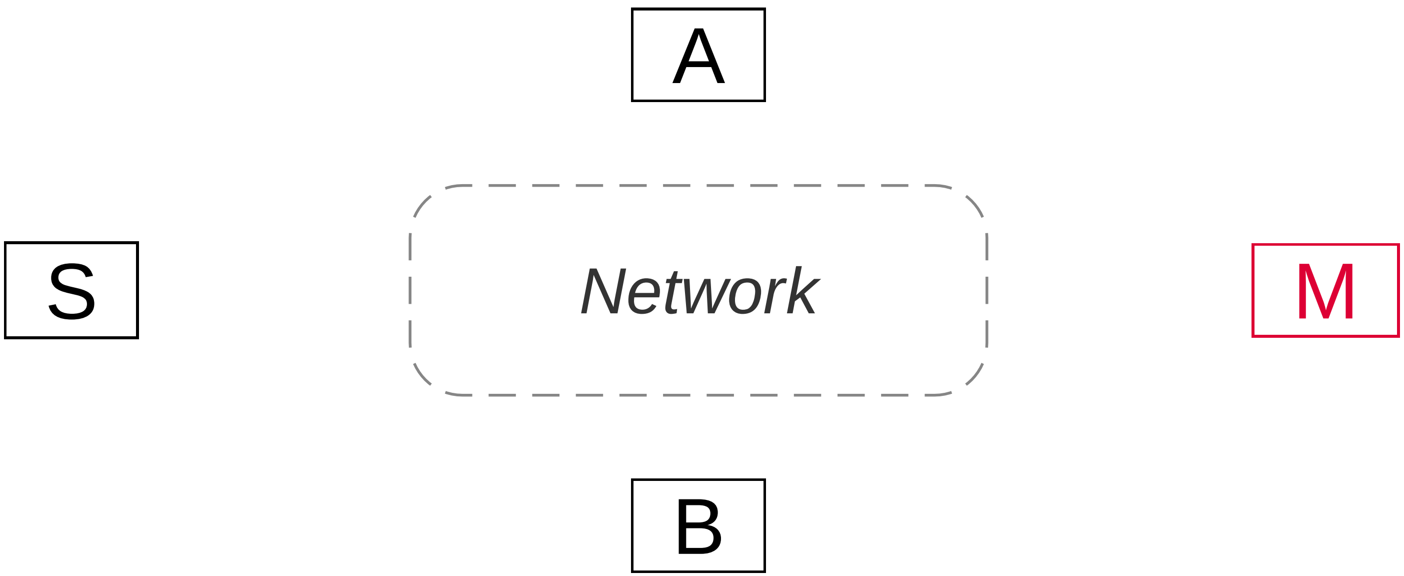
Example Revisited

$$\begin{aligned} A \rightarrow S: &\quad A, B, \{K\}_{K_A} \\ S \rightarrow B: &\quad A, \{K\}_{K_B} \\ A \rightarrow B: &\quad \{m_1, \cdots, m_k\}_K \end{aligned}$$

The analysis of n instances of the protocol gives

$$\psi = \{\ (A_i, B_j), (A_i, \ell_\bullet), (\ell_\bullet, B_j) \mid 1 \leq i, j \leq n\}$$

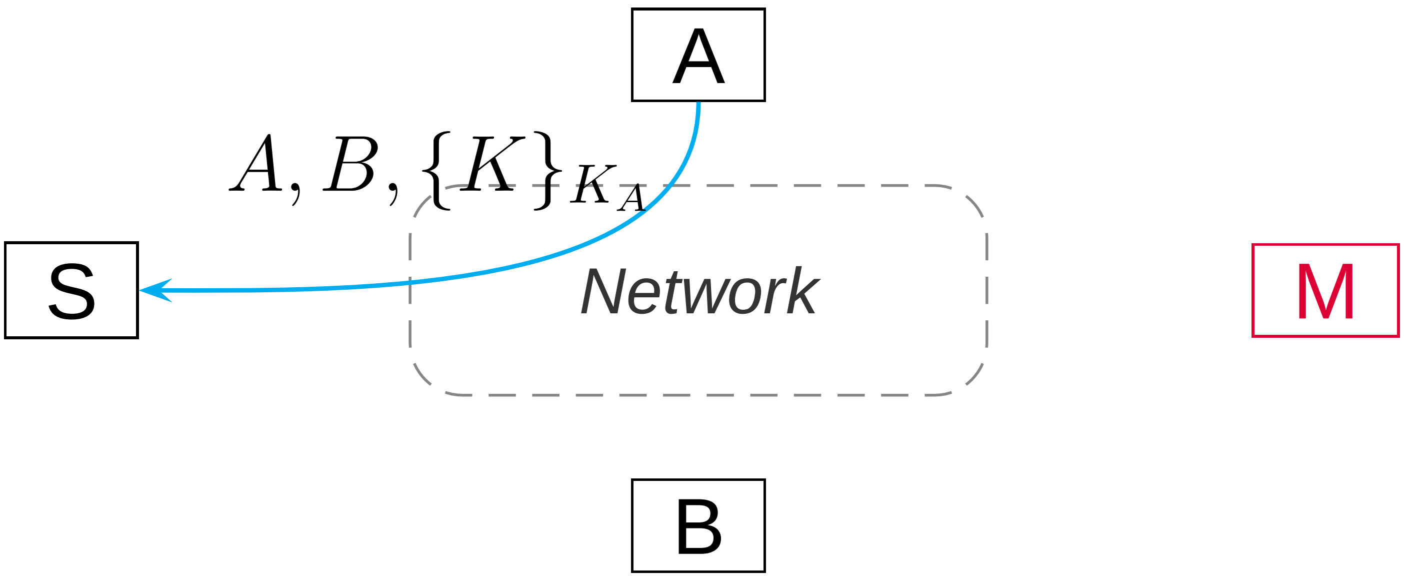
An Attack



Flawed protocol inspired by "Wide Mouthed Frog"

1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$

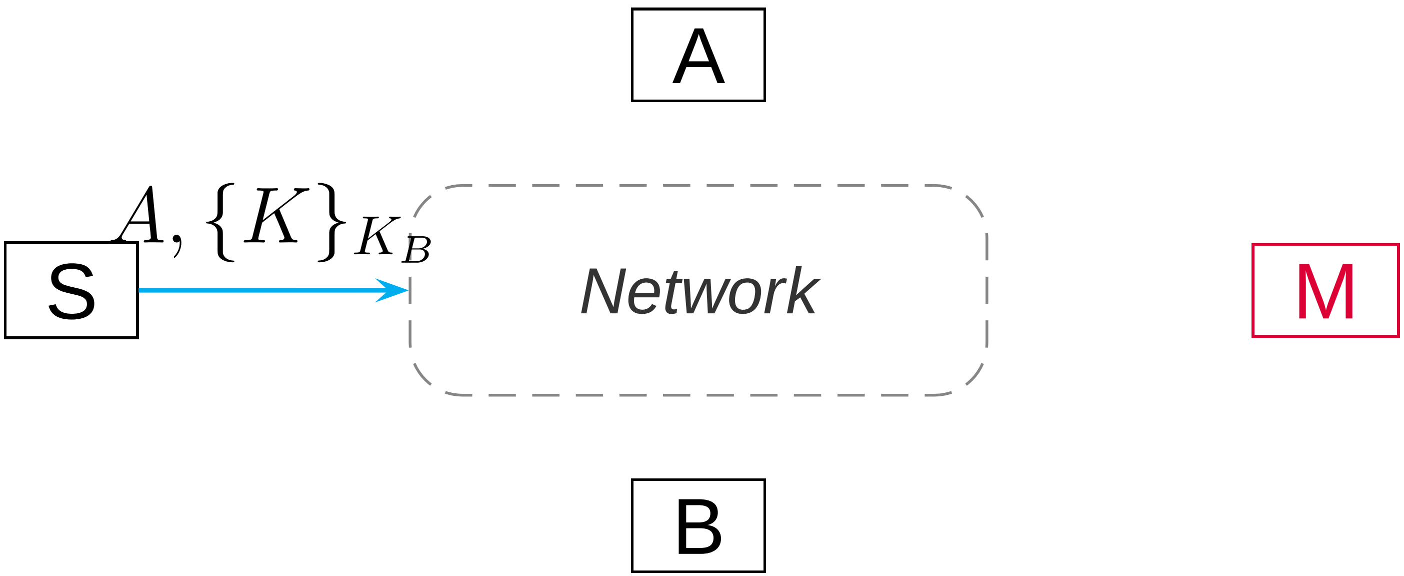
An Attack



Flawed protocol inspired by "Wide Mouthed Frog"

1. $A \rightarrow S: \quad A, B, \{K\}_{K_A}$ $\qquad A \rightarrow S: \quad A, B, \{K\}_{K_A}$
2. $S \rightarrow B: \quad A, \{K\}_{K_B}$
3. $A \rightarrow B: \quad \{m_1, \cdots, m_k\}_K$

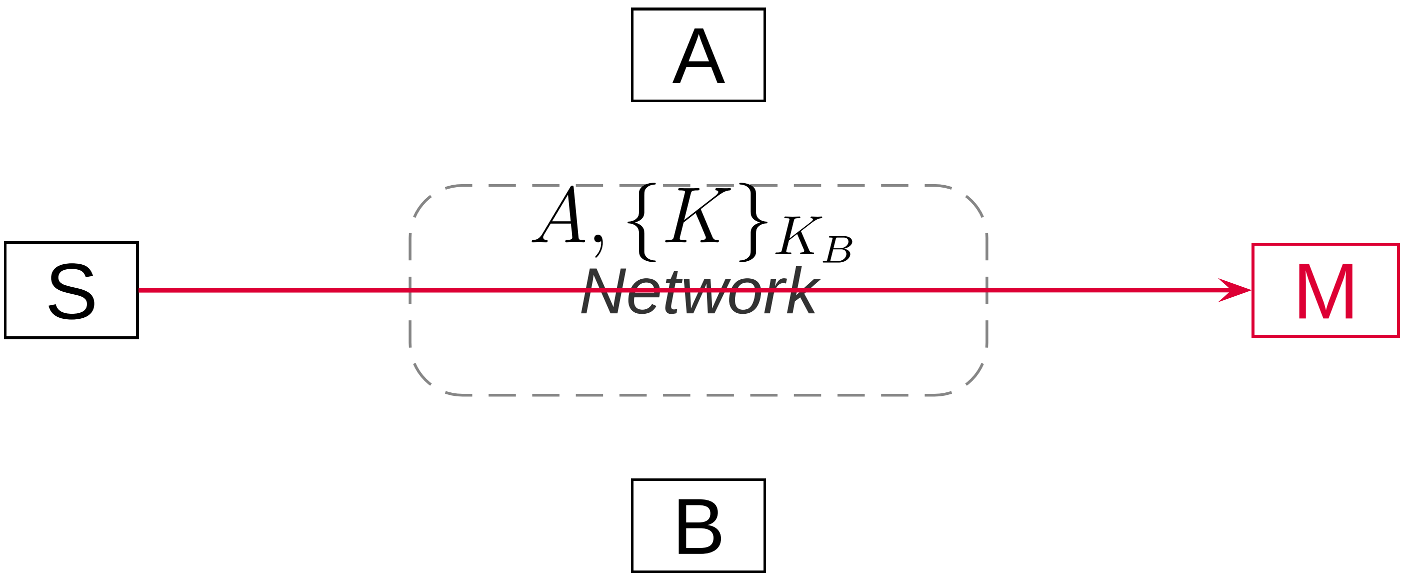
An Attack



Flawed protocol inspired by “Wide Mouthed Frog”

$$
\begin{array}{llll}
1. & A \rightarrow S : & A, B, \{K\}_{K_A} & \quad A \rightarrow S : \quad A, B, \{K\}_{K_A} \\
2. & S \rightarrow B : & A, \{K\}_{K_B} & \\
3. & A \rightarrow B : & \{m_1, \cdots, m_k\}_K &
\end{array}
$$

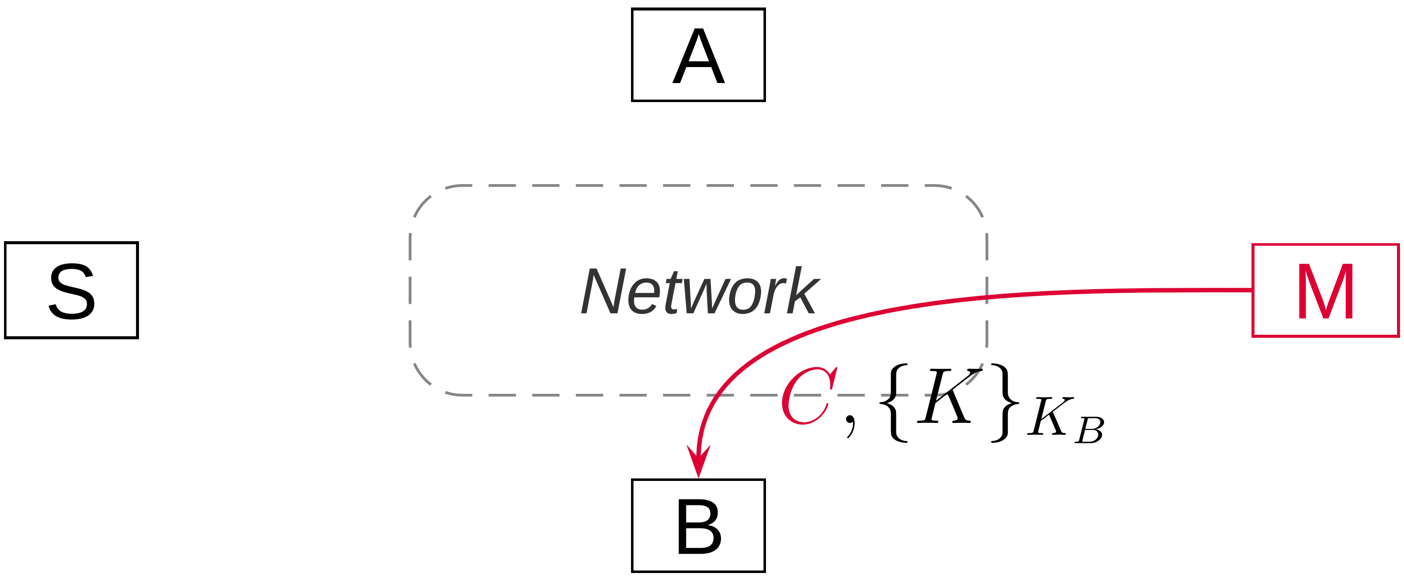
An Attack



Flawed protocol inspired by “Wide Mouthed Frog”

$$
\begin{array}{llll}
1. & A \rightarrow S: & A, B, \{K\}_{K_A} & A \rightarrow S: & A, B, \{K\}_{K_A} \\
2. & S \rightarrow B: & A, \{K\}_{K_B} & S \rightarrow M(B): & A, \{K\}_{K_B} \\
3. & A \rightarrow B: & \{m_1, \cdots, m_k\}_K & &
\end{array}
$$

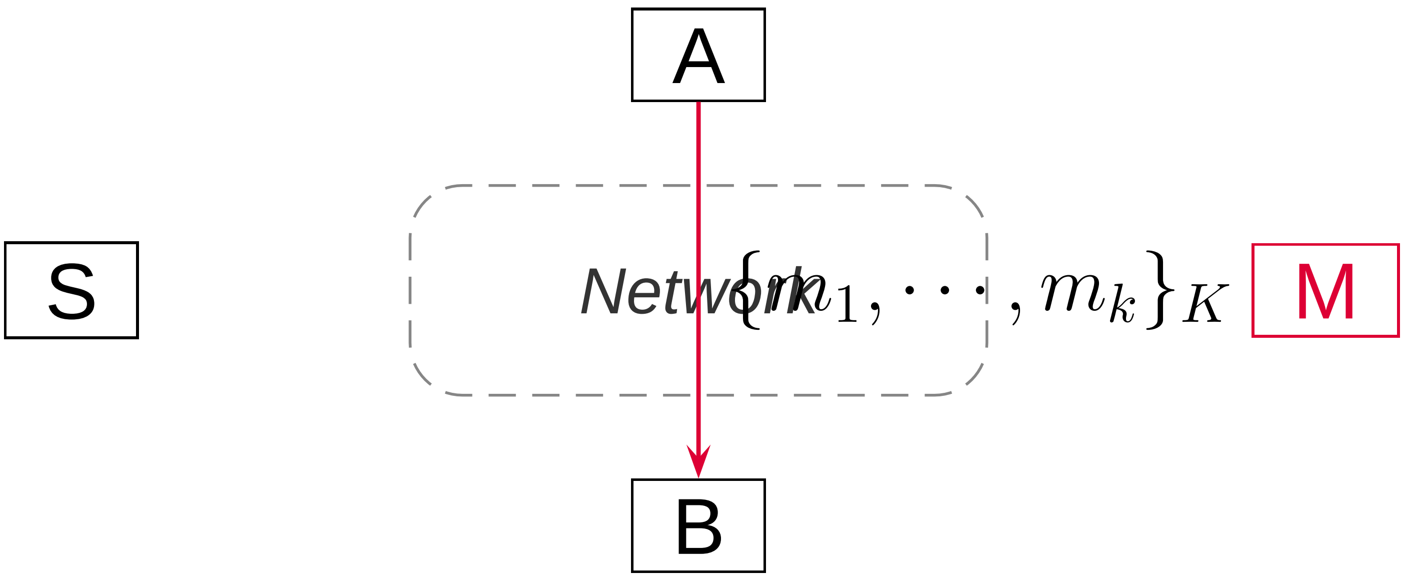
An Attack



Flawed protocol inspired by "Wide Mouthed Frog"

1.	$A \to S :$	$A, B, \{K\}_{K_A}$	$A \to S :$	$A, B, \{K\}_{K_A}$
2.	$S \to B :$	$A, \{K\}_{K_B}$	$S \to M(B) :$	$A, \{K\}_{K_B}$
3.	$A \to B :$	$\{m_1, \cdots, m_k\}_K$	$M(S) \to B :$	$C, \{K\}_{K_B}$

An Attack



Flawed protocol inspired by “Wide Mouthed Frog”

1.	$A \to S:$	$A, B, \{K\}_{K_A}$	$A \to S:$	$A, B, \{K\}_{K_A}$
2.	$S \to B:$	$A, \{K\}_{K_B}$	$S \to M(B):$	$A, \{K\}_{K_B}$
3.	$A \to B:$	$\{m_1, \cdots, m_k\}_K$	$M(S) \to B:$	$C, \{K\}_{K_B}$
			$A \to B:$	$\{m_1, \cdots, m_k\}_K$

Example Revisited

$$\begin{array}{ll} A \rightarrow S: & A, B, \{K\}_{K_A} \\ S \rightarrow B: & A, \{K\}_{K_B} \\ A \rightarrow B: & \{m_1, \cdots, m_k\}_K \end{array}$$

$$\begin{array}{ll} A \rightarrow S: & A, B, \{K\}_{K_A} \\ S \rightarrow M(B): & A, \{K\}_{K_B} \\ M(S) \rightarrow B: & C, \{K\}_{K_B} \\ A \rightarrow B: & \{m_1, \cdots, m_k\}_K \end{array}$$

The analysis of n instances of the protocol gives

$$\psi = \{\ (A_i, B_j), (A_i, \ell_\bullet), (\ell_\bullet, B_j) \mid 1 \leq i, j \leq n\}$$

Analysis of Standard Protocols

Our analysis identifis a number of authentication flaws in symmetric key protocols such as Needham-Schroeder, Otway-Rees, Yahalom and Andrew Secure RPC. It shows that

- many classical problems occur precisely because some crucial distinctions (between identities and roles) are not sufficiently clear.
- Many protocols become insecure when old session keys are compromised.

General Considerations (1)

- The analysis identifies the well-known attacks on the protocols we have considered
- When well-known amendments are performed the analysis reports that there are no attacks
- Very few false positives
- Polynomial time validation procedure
- Improve the precision of the analysis

General Considerations (2)

The same approach is valid also for

- Other cryptographic features, such as asymmetric cryptography
- Other security properties (by checking other annotations): e.g. on freshness and type flaws, ...
- Other calculi: variations of LySa, Beta Binders, ...

Asymmetric Cryptography

$E ::=$	$n / (m^+, m-)$ key/key pair
$\{E_1, \cdots, E_k\}_{E_0}$	symm. encryption
$\{\!\mid E_1, \cdots, E_k \mid\!\}_{E_0}$	asymm. encryption
$P ::=$	...
$\langle E_1, \cdots, E_k \rangle . P$	output
$(E_1, \cdots, E_j; x_{j+1}, \cdots, x_k). P$	input (with matching)

decrypt E as $\{E_1, \cdots, E_j; x_{j+1}, \cdots, x_k\}_{E_0}$ in P

symm. decryption (with matching)

decrypt E as $\{\!\mid E_1, \cdots, E_j; x_{j+1}, \cdots, x_k \mid\!\}_{E_0}$ in P

asymm. decryption (with matching)

$(\nu n)/(\nu m^+, m^-)P$	key/key pair creation

Considerations

- Our CFA (and the related tool) is able to deal with asymmetric cryptography
- The analysis identifies the well-known attacks on the protocols considered (e.g. Lowe's one on Needham Schroeder PK)
- The analysis discover an undocumented flow in the Beller-Chang-Jacobi MSR protocol

Some References (1)

BDNN01 C.Bodei, P.Degano, F.Nielson and H. Riis Nielson. Static Analysis for the Pi-Calculus with Applications to Security. Information and Computation, 168: 68-92, 2001.

BDNN02 Bodei, Degano, Nielson, Riis Nielson. Flow Logic for Dolev-Yao Secrecy in Cryptographic Processes, FGCS 18, 2002.

BBDNN03 Bodei C., Buchholtz M., Degano, P., Nielson, F. & Riis Nielson, H. Automatic Validation of Protocol Narration, Proc. CSFW'03.

BDP05 Bodei, C., Degano, P., Priami, C. Checking Security Policies through an Enhanced Control Flow Analysis. Journal of Computer Security, 13(1): 49-85, 2005.

Some References (2)

BBDNN05 Bodei C., Buchholtz M., Degano, P., Nielson, F. & Riis Nielson, H. Static Validation of Security Protocols. Journal of Computer Security, 13(3): 347-390, 2005.

GBDN07 Gao H., Bodei C., Degano, P. & Riis Nielson H. A Formal Analysis for Capturing Replay Attacks in Cryptographic Protocols. Proc. of ASIAN'07. To appear in LNCS, 2007.

BDGB07 Bodei C., Degano, P., Gao H. & Brodo L. Detecting and Preventing Type Flaws: a Control Flow Analysis with tags. Proc. of SecCO'07. To appear in ENTCS, 2007.