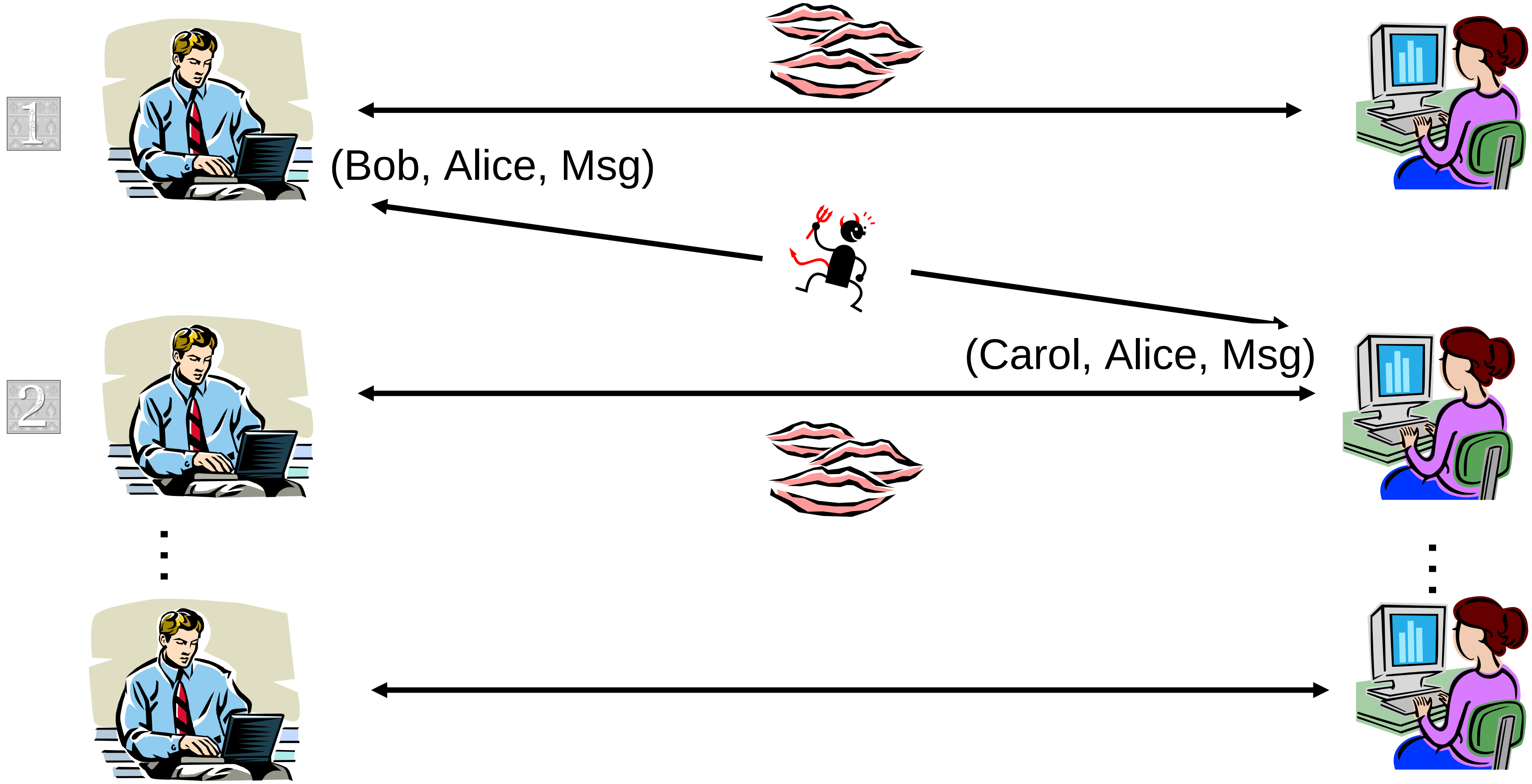
A Formal Analysis for Capturing Replay Attacks in Cryptographic Protocols

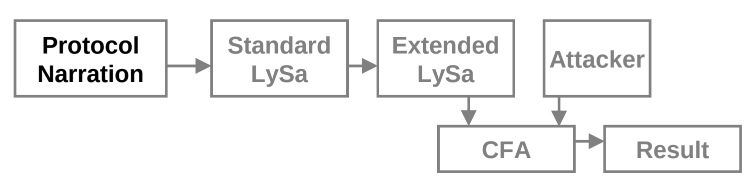
Han Gao[1], Chiara Bodei[2] ,
Pierpaolo Degano[2], Hanne Riis Nielson[1]

Informatics and Mathematics Modelling,
Technical University of Denmark[1]
Dipartimento di Informatica, Università di Pisa[2]

ASIAN'07 Doha, December 2007

Replay Attacks in Protocols





Needham-Schroeder

Flaw discovered in 1981

- Invented in 1978

1. $A \rightarrow S: \quad A, B, N_a$
2. $S \rightarrow A: \quad \{N_a, B, K, \{K, A\}_{K_b}\}_{K_a}$
3. $A \rightarrow B: \quad \{A, K\}_{K_b}$
4. $B \rightarrow A: \quad \{N_b\}_K$
5. $A \rightarrow B: \quad \{N_b - 1\}_K$
6. $A \rightarrow B: \quad \{Msg\}_K$

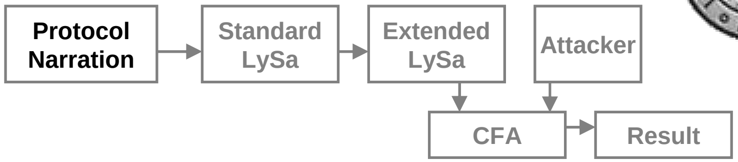
Key distribution steps: (steps 1–3)

The key should be known to both A and B

Authentication steps: (steps 4–5)

A and B make sure that they both know the key

Message exchange step (step 6)



Needham-Schroeder

An old session key K' is leaked

- The Denning-Sacco Attack

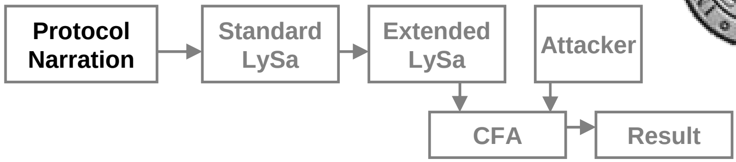
$$
\begin{array}{lll}
1. & A \to S: & A, B, N_a \\
2. & S \to A: & \{N_a, B, K, \{K, A\}_{K_b}\}_{K_a} \\
3. & A \to B: & \{A, K\}_{K_b} \\
4. & B \to A: & \{N_b\}_K \\
5. & A \to B: & \{N_b - 1\}_K \\
6. & A \to B: & \{Msg\}_K
\end{array}
$$

$$
\begin{array}{lll}
1. & \ldots & \\
2. & \ldots & \\
3. & M(A) \to B: & \{A, K'\}_{K_b} \\
4. & B \to M(A): & \{N_b\}_{K'} \\
5. & M(A) \to B: & \{N_b - 1\}_{K'} \\
6. & M(A) \to B: & \{Msg\}_{K'}
\end{array}
$$

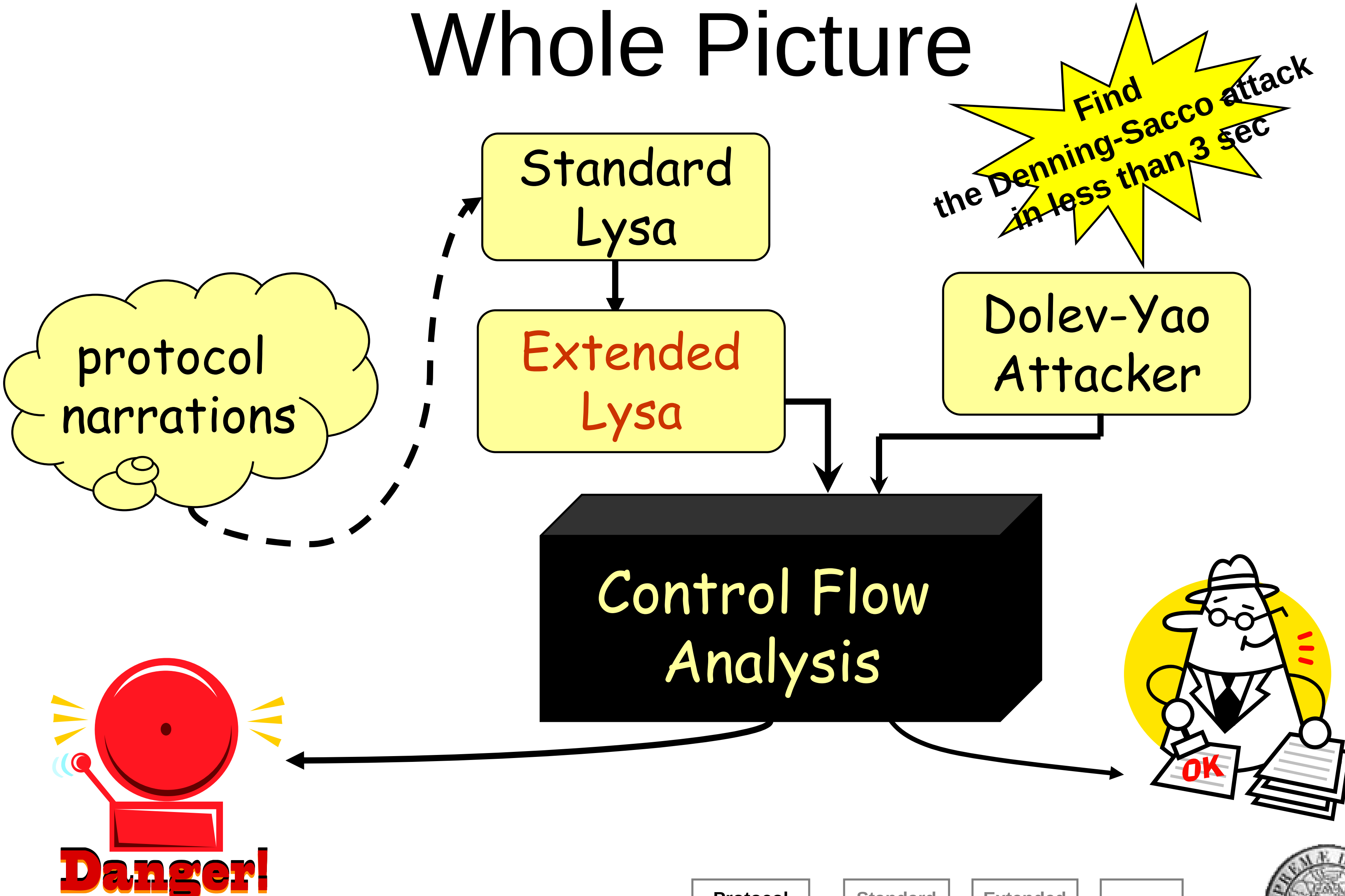
A is convinced that K is fresh

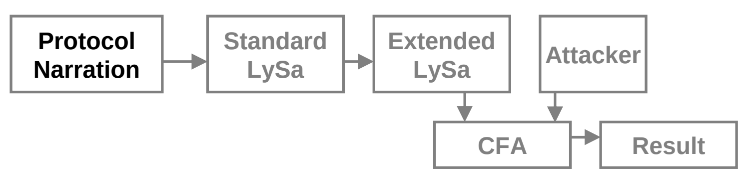
B believes he is talking to A!

No such guarantee for B



Whole Picture





LySa Calculus

One global channel

1. $A \rightarrow S : \quad A, B, N_a$
2. $S \rightarrow A : \quad \{N_a, B, K, \{K, A\}_{K_b}\}_{K_a}$
3. $A \rightarrow B : \quad \{A, K\}_{K_b}$
4. $B \rightarrow A : \quad \{N_b\}_K$
5. $A \rightarrow B : \quad \{N_b - 1\}_K$
6. $A \rightarrow B : \quad \{Msg\}_K$

sender, receiver, payload:

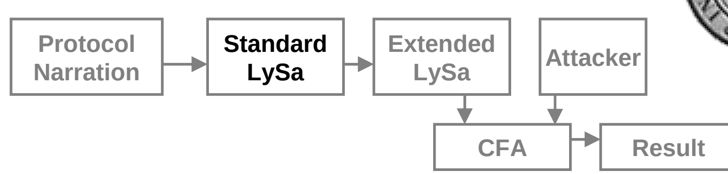
$\langle A, S, A, B, N_a \rangle.$

$\langle A, B, \{A, K\}_{K_b} \rangle.$

pattern matching, variable binding:

$(A, B; y).$
$decrypt\ y\ as\ \{A; k\}_{K_b}\ in$
$\ldots$

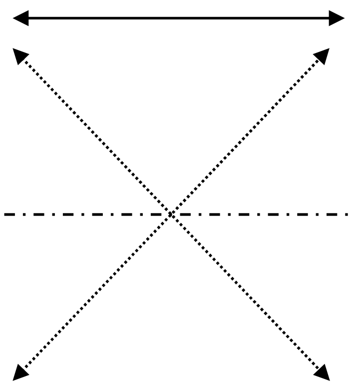
$$P = P_A \mid P_B \mid P_S$$



Session Identifiers

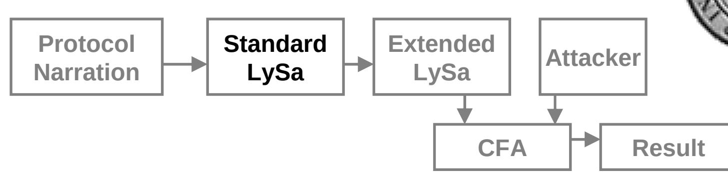
protocol run 1

$\langle A, S, N_a \rangle.$ $(A, S; x).$



protocol run 2

$\langle A, S, N_a \rangle.$ $(A, S; x).$



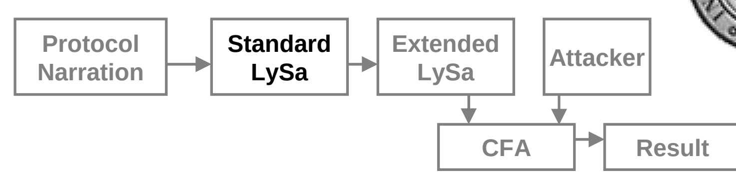
Session Identifiers

protocol run 1

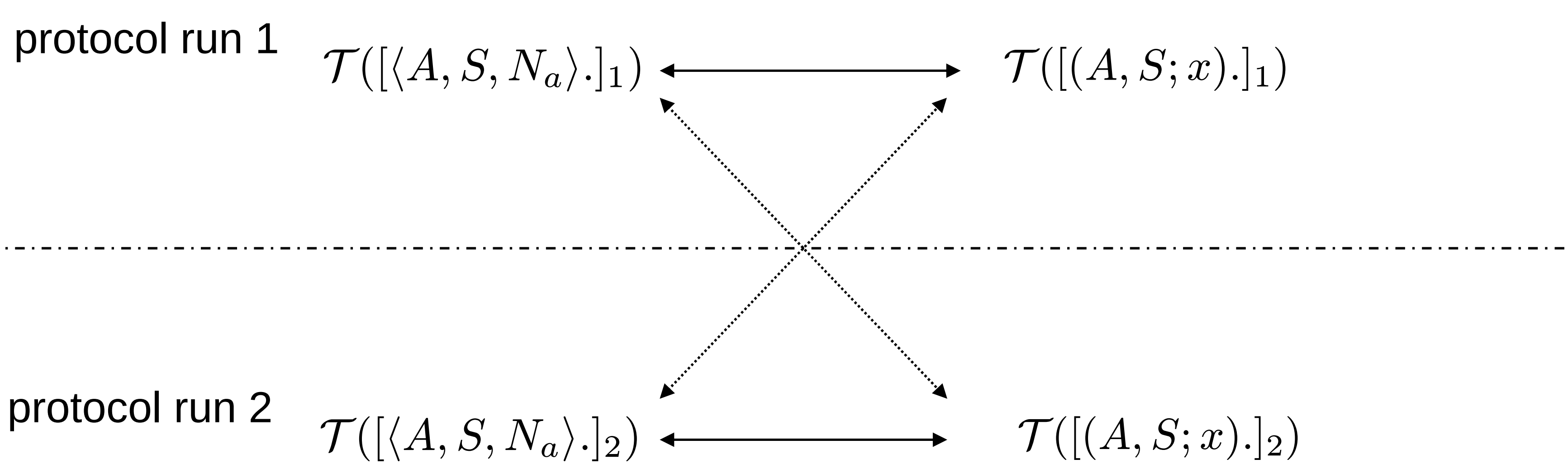
$[\langle A, S, N_a \rangle .]_1$ ⟷ $[(A, S; x).]_1$

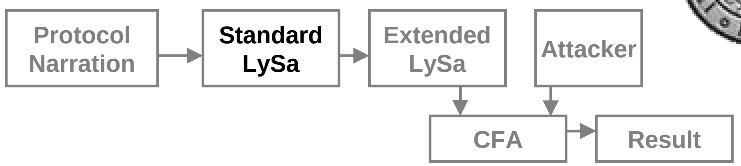
protocol run 2

$[\langle A, S, N_a \rangle .]_2$ ⟷ $[(A, S; x).]_2$



Session Identifiers





Extended LySa Calculus

Stops when reaching n or x

	Standard		Extended
Terms	E	$\xrightarrow{\mathcal{F}}$	$\mathcal{E}$
Processes	P	$\xrightarrow[\mathcal{T}]{}$	$\mathcal{P}$

Stops when reaching 0 or !

$$\mathcal{F}([\{N\}_K]_s) = \{[N]_s\}_{[K]_s}$$

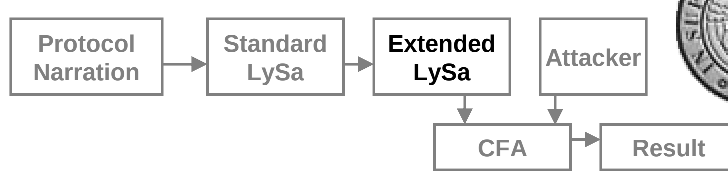
$$\mathcal{T}([\langle N\rangle.0 \mid !((;x).0)]_s) =$$
$$\mathcal{T}([\langle N\rangle.0]_s) \mid \mathcal{T}([!((;x).0)]_s) =$$
$$\langle [N_s]\rangle.0 \mid [!((;x).0)]_s$$

1. $A \rightarrow S: \quad A, B, N_a$ — $\langle A, S, A, B, N_a\rangle.$
2. $S \rightarrow A: \quad \{N_a, B, K, \{K, A\}_{K_b}\}_{K_a}$
3. $A \rightarrow B: \quad \{A, K\}_{K_b}$ — $\langle A, B, \{A, K\}_{K_b}\rangle.$ — $(A, B; y).$ *decrypt* y *as* $\{A; k\}_{K_b}$ *in* $\ldots$
4. $B \rightarrow A: \quad \{N_b\}_K$
5. $A \rightarrow B: \quad \{N_b - 1\}_K$
6. $A \rightarrow B: \quad \{Msg\}_K$

$$P = P_A \mid P_B \mid P_S$$

$$\mathcal{P} = [!P]_0$$

Unfold once in each semantics step



Freshness Property

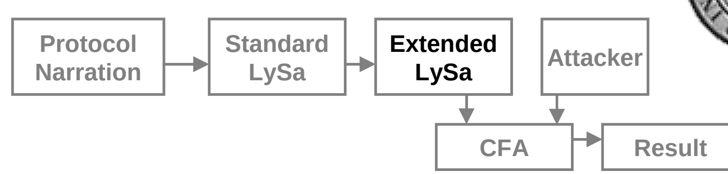
Equality with sessin IDs ingnored

Extract the session ID

$$\frac{\mathcal{E}_0 \approx \mathcal{E}'_0 \;\wedge\; \mathcal{E}_1 \approx \mathcal{E}'_1 \;\wedge\; \mathcal{R}(\mathcal{I}(\mathcal{E}_0), \mathcal{I}(\mathcal{E}'_0)) \wedge \mathcal{R}(\mathcal{I}(\mathcal{E}_1), \mathcal{I}(\mathcal{E}'_1))}{decrypt\ [\{\mathcal{E}_1, \mathcal{E}_2\}_{\mathcal{E}_0}]_s\ as\ \{\mathcal{E}'_1; x_2\}_{\mathcal{E}'_0}\ in\ \mathcal{P} \rightarrow_{\mathcal{R}} \mathcal{P}[\mathcal{E}_2/x_2]}$$

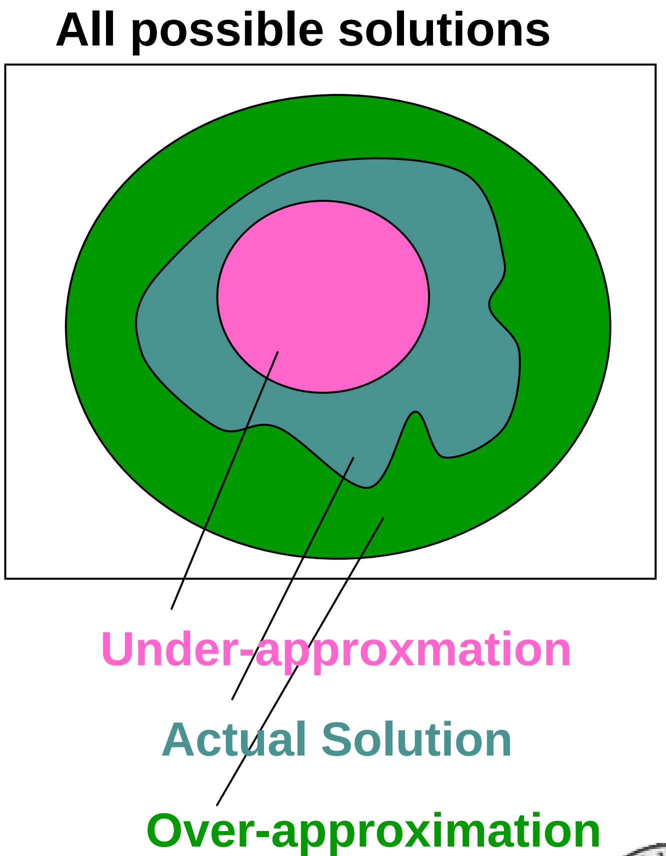
$$decrypt\ \{[N_a]_1, [N_b]_1\}_{[K]_1}\ as\ \{[N_a]_1; x\}_{[K]_1}\ in\ 0$$

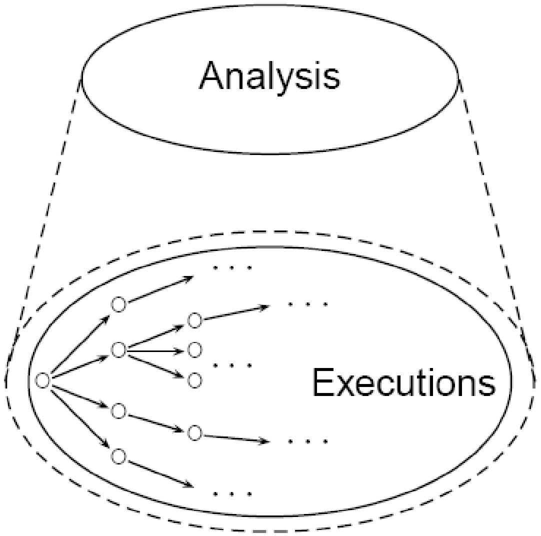
$$decrypt\ \{[N_a]_2, [N_b]_2\}_{[K]_2}\ as\ \{[N_a]_1; x\}_{[K]_1}\ in\ 0$$

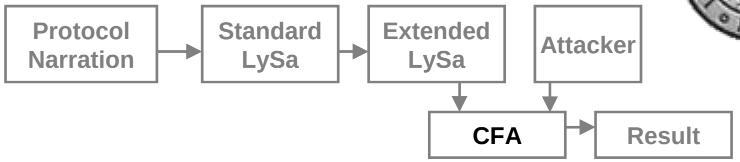


Static Analysis

- Approximation
  - Over-Approximation
- Algorithms
  - Control Flow Analysis



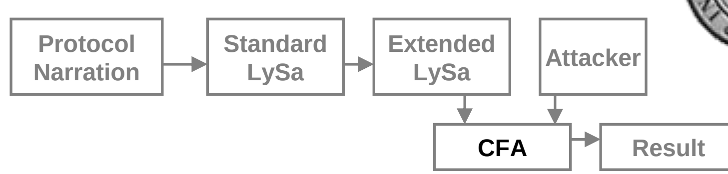




Static Analysis

- Analysis of Terms $\rho \models \mathcal{E} : \vartheta$
  - Determine the possible values that each term may evaluate to
- Analysis of Processes $\rho, \kappa \models_{\text{RM}} \mathcal{P} : \psi$
  - Collect the values that may flow on the network
  - Error component

$$\frac{\text{analysis}(\mathcal{T}(\lfloor P \rfloor_0)) \mid \text{analysis}(\mathcal{T}(\lfloor P \rfloor_1))}{\text{analysis}(\mathcal{P})}$$

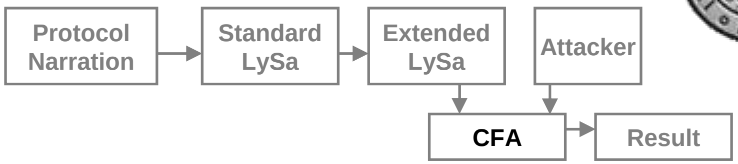


The Error Component

- The error component ψ collects labels of decryption where freshness violations may happen. For example:

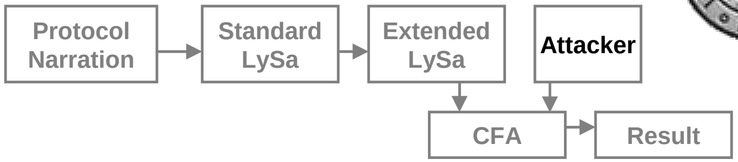
$$l \in \psi$$

- The empty error component implies free of replay attacks at run time



The Attacker

- Capabilities
  - Eavesdrop
  - Alter
  - Insider or outsider or both
  - Obtain old session keys



Analysis of Needham-Schroeder

1. $A \to S: \quad A, B, N_a$
2. $S \to A: \quad \{N_a, B, K, \{K, A\}_{K_b}\}_{K_a}$
3. $A \to B: \quad \{A, K\}_{K_b}$
4. $B \to A: \quad \{N_b\}_K$
5. $A \to B: \quad \{N_b - 1\}_K$
6. $A \to B: \quad \{Msg\}_K$

$\langle A, B, \{A, K\}_{K_b} \rangle.$

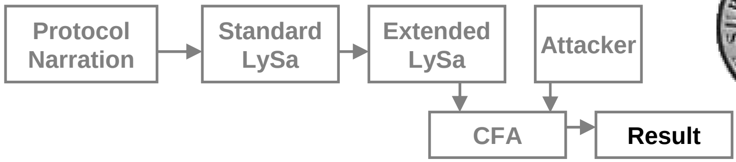
$(A, B; y).$
$decrypt\ y\ as\ \{A; k\}_{K_b}\ in$
$\ldots$

$$P = P_A \mid P_B \mid P_S$$
$$\mathcal{P} = [!P]_0$$

$$\frac{\text{analysis}(\mathcal{T}([P]_0)) \mid \text{analysis}(\mathcal{T}([P]_1))}{\text{analysis}(\mathcal{P})}$$

0 $\mathcal{T}([\langle A, B, \{A, K\}_{K_b} \rangle]_0) \longrightarrow \mathcal{T}([(A, B, y).\ decrypt\ y\ as\ \{A; k\}_{K_b}\ in]_0)$

Session 1 $\mathcal{T}([\langle A, B, \{A, K\}_{K_b} \rangle]_1) \longrightarrow \mathcal{T}([(A, B, y).\ decrypt\ y\ as\ \{A; k\}_{K_b}\ in]_1)$



Conclusion

- Simply process calculus with cryptographic primitives for modelling security protocols
- Automatic algorithm for providing security assurances for protocols
  - Semantics correct and sound
- Implementation has been used to validate a number of protocols

Thank You!

The Control Flow Analysis

- Over-approximate the protocol behaviour
- The values of the variables

$$\rho : X \to \mathcal{P}(Val)$$

- The messages flowing on the network

$$\kappa \subseteq \mathcal{P}(Val^*)$$

- For example:

$$\langle [A]_1, [B]_1, [N]_1 \rangle \in \kappa$$

$$[N]_1 \in \rho(x)$$

Judgement for Decryption

- At each decryption point, check whether freshness may be violated

$\rho \models \mathcal{E} : \vartheta \ \wedge \ \mathcal{E}_1 : \vartheta_1 \ \wedge$	evaluate terms
$\rho \models \mathcal{E}_0 : \vartheta_0 \ \wedge$	evaluate key
$\forall [\{v_1, v_2\}_{v_0}]_s \in \vartheta : v_0 \propto \vartheta_0 \wedge$	for all encrypted values
$v_1 \propto \vartheta_1 \Rightarrow$	pattern matching
$v_2 \in \rho(x_2) \wedge$	variable binding
$(\mathcal{I}(v_1) \neq \mathcal{I}(\mathcal{E}_1) \Rightarrow l \in \psi) \ \wedge$	freshness checking
$\rho, \kappa \models \mathcal{P} : \psi$	analyse the rest

$$\frac{\rho \models \mathcal{E} : \vartheta \ \wedge \ \mathcal{E}_1 : \vartheta_1 \ \wedge \quad \rho \models \mathcal{E}_0 : \vartheta_0 \ \wedge \quad \forall [\{v_1, v_2\}_{v_0}]_s \in \vartheta : v_0 \propto \vartheta_0 \wedge \quad v_1 \propto \vartheta_1 \Rightarrow v_2 \in \rho(x_2) \wedge (\mathcal{I}(v_1) \neq \mathcal{I}(\mathcal{E}_1) \Rightarrow l \in \psi) \ \wedge \ \rho, \kappa \models \mathcal{P} : \psi}{\rho, \kappa \models \mathit{decrypt}\ \mathcal{E}\ \mathit{as}\ \{\mathcal{E}_1; x_1\}^l_{\mathcal{E}_0}\ \mathit{in}\ \mathcal{P} : \psi}$$

$\propto$: membership relation with session IDs ignored