Tools for Verification of Security Protocols

Petr Matoušek

matousp@fit.vutbr.cz

Brno University of Technology, Czech republic

Introduction

- **Talk Outline**

1. **Security protocols – motivation**
2. **LySa tool – static analysis of security protocols**
   - **Describing protocol in** LYSA
   - LYSA **process calculus**
   - **Control Flow Analysis**
3. **Model Checking Security Protocols using OFMC**
   - **Modeling Protocol Behavior**
   - **Formal Protocol Analysis using Model Checking (MC)**
   - **OFMC and AVISPA project**
4. **References**

1. Security Protocols

❖ What is the problem?

- **A wants to communicate in a secure way with B over insecure medium.**

❖ What could go wrong?

- **interruption, eavesdropping, modification, traffic analysis, fake data**

❖ What do we want?

- **confidentiality**
- **integrity**
- **authentication**
- **non-repudiation**
- **availability**

1. Security Protocols

❖ Security Protocols

- a set of rules that describes the exchange of messages between two or more principals
- security protocols uses cryptographic mechanisms to achieve security objectives

❖ Security mechanisms

- authentication, key establishment, timeliness, session keys
- symmetric cryptography
- asymmetric cryptography
- signatures, hashes

1. Security Protocols

❖ Protocol analysis

- **difficult to specify properties**
- **protocols often described informally**
- **all possible attacks should be treated**

❖ Approaches

- **static analysis –** LYSA**, process's approach**
- **model checking – special tools OFMC, traces**
- **model checking – general tools:** UPPAAL**,** PRISM
- **Colored Petri Nets – CPN tool, etc.**

❖ Inspiration

- **Verification of Protocols for Security and Mobility (VPSM) 2006**
- **see** `http://www.first.dk/VPSM`

1. Security Protocols – Example

- **Needham-Schroeder Symmetric Key Protocol**
  - **defined in 1978**
  - **two parties (A,B) trying to communicate with a session key given by S**
- **Communication**

  1. $A \rightarrow S : A, B, N_a$

  2. $S \rightarrow A : E[K_A](N_a, B, K, E[K_B](K, A))$

  3. $A \rightarrow B : E[K_B](K, A)$

  4. $B \rightarrow A : E[K](N_b)$

  5. $A \rightarrow B : E[K](N_{b-1})$

- **Is this protocol design correct in terms of security?**

1. Security Protocols – Example

❖ Needham-Schroeder Symmetric Key Protocol

1. $A \rightarrow S : A, B, N_a$
2. $S \rightarrow A : E[K_A](N_a, B, K, E[K_B](K, A))$
3. $A \rightarrow B : E[K_B](K, A)$
4. $B \rightarrow A : E[K](N_b)$
5. $A \rightarrow B : E[K](N_{b-1})$

❖ Denning-Sacco Attack, 1981

1. $A \rightarrow S : A, B, N_a$
2. $S \rightarrow A : E[K_A](N_a, B, K, E[K_B](K, A))$

- **leaking the key $\rightarrow$ the intruder $I(A)$ captures old session key K' and message $E[K_B](K', A)$**

3. $A \rightarrow I(A) : E[K_B](K, A)$
3. $I(A) \rightarrow B : E[K_B](K', A)$
4. $B \rightarrow I(A) : E[K'](N_b)$
5. $I(A) \rightarrow B : E[K'](N_{b-1})$

- **B believes he is talking with A**

1. Security Protocols – Example

❖ **What we need in order to validate a security protocol?**

- **unambiguous and complete description**

  ⇒ **protocol description with well-defined semantics**
- **the assumptions under which protocol operates is clear**

  ⇒ **formal specification**
- **ensure that the protocol fulfils security goal under given assumptions**

  ⇒ **formal validation**

2. LYSA tool – static analysis

2.1 Protocol description in LYSA – Wide Mouthed Frog (WMF) protocol

- **secret (symmetric) session key K between two principals A and B**
- **A and B shares master keys K_A and K_B with a trusted server S**

Scenario

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

Three communicating processes

1. A **– creates a new key, sends the key to S, sends messages to B**
2. B **– receives the key by S, decrypts the message, receives the mesage by A, decrypts the message**
3. S **– receives the key by A, decrypt the key, sends the encrypted message to B**

2. LySa tool – WMF in LySa

❖ Scenario

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

❖ LySa description

1. $(\nu\ K)$
1. $< A, S, A, \{B, K\}_{K^A} > .$
1. $(\nu\ m) < A, B, \{m\}_K > .\ 0$
2. $|\ (S, B; y).$
2. **decrypt** y **as** $\{A; y^K\}_{K^B}$ **in**
2. $(A, B; z).$
2. **decrypt** z **as** $\{; z^m\}_{y^K}$ **in** 0
3. $|\ (A, S, A; x).$
3. **decrypt** x **as** $\{B; x^K\}_{K^A}$ **in**
3. $< S, B, \{A, x^K\}_{K^B} > .0$

2. LySa tool – WMF in LySa

❖ Scenario

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

❖ LySa description – adding assumptions (encryptions)

1. $(\nu\ K)$

1. $\quad < A, S, A, \{B, K\}^{A}_{K^A}$ $[$**dest** $S] > .$

1. $\quad (\nu\ m) < A, B, \{m\}^{A}_{K}$ $[$**dest** $B] > .\ 0$

2. $\mid (S, B; y).$

2. $\quad$ **decrypt** y **as** $\{A; y^K\}^{B}_{K^B}$ $[$**orig** $S]$ **in**

2. $\quad (A, B; z).$

2. $\quad$ **decrypt** z **as** $\{; z^m\}^{B}_{y^K}$ $[$**orig** $A]$ **in** 0

3. $\mid (A, S, A; x).$

3. $\quad$ **decrypt** x **as** $\{B; x^K\}^{S}_{K^A}$ $[$**orig** $A]$ **in**

3. $\quad < S, B, \{A, x^K\}^{S}_{K^B}$ $[$**dest** $B] > .0$

2. LYSA–calculus

❖ LYSA–calculus

- a process algebra
- based on CCS, π–calculus, and Spi-calculus
- supports massive parallelism
- incorporate communication
- handle cryptographic primitives
- can be extended to handle mobility and locations
- have formal semantics
- is subject to automatic analysis
- supports only one transmission channel – the ether

2. LYSA–calculus

❖ LYSA–syntax

- **process**

P	::=	0	**terminated process**
	|	$P1 \mid P2$	**parallel processes**
	|	$!\ P$	**replication**
	|	$(\nu\ n)\ P$	**introduction of a new name in the scope P**
	|	$< E_1, \ldots, E_k > .P$	**output to the ether**
	|	$(x_1, \ldots, x_k).P$	**input from the ether**
	|	**decrypt** E **as** $\{E_1, \ldots, E_j; x_{j+1}, \ldots, x_k\}_{E_0}$ **in** P **– symmetric decryption**	

- **expression**

E	::=	n	**name**
	|	x	**variable**
	|	$\{E_1, \ldots, E_k\}_{E_0}$	**symmetric encryption**

2. LYSA–calculus

❖ LYSA–syntax

- **assertions for origin and destinations**

$\{E_1, \ldots, E_k\}^l_{E_0}$ [**dest** $\mathcal{L}$]	**encryption**
decrypt E **as** $\{E'_1, \ldots, E'_j; x_{j+1}, \ldots, x_k\}^l_{E'_0}$ [**orig** $\mathcal{L}$] **in** P	**symmetric decryption**

❖ LYSA–semantics

- **communication rule** $\dfrac{[E_1]=[E'_1]}{<E_1,E_2>.P \mid (E'_1;x_2).Q \rightarrow P \mid Q[E_2/x_2]}$
- **decryption rule**
- **parallel rule**
- **reduction rule**
- **structural congruence**

2. LySa–Control Flow Analysis

❖ Static Program Analysis

- the aim is to efficiently compute safe approximations to the behaviour of programs without running them
- constraint based technique
- inherits methodology from type systems:
  - specification
  - semantic properties
  - algorithmic realization
  - judgements
  - subject reduction
  - solver technology

2. LySa–Control Flow Analysis

❖ The idea behind the analysis

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

❖ Example

1. $< A, S, A, \{B, K\}^{A}_{K^A}[\textbf{dest } S] > \ldots$

2. $\mid (A, S, A; x).$

3. **decrypt** x **as** $\{B; x^K\}^{S}_{K^A} \ldots$

$\Rightarrow \; < A, S, A, \{B, K\}^{A}_{K^A}[\textbf{dest } S] > \in \kappa$

$\Rightarrow \; \{B, K\}^{A}_{K^A}[\textbf{dest } S] \in \rho(x)$

$\Rightarrow \; K \in \rho(x^K)$

2. LySa–Control Flow Analysis

- **Judgements of the analysis**
  - **for terms:** $\rho \models E : \vartheta$
    - **estimation of the set of values ϑ that E may evaluate to in the context given by ρ**
  - **for processes** $(\rho, \kappa) \models_{RM} P : \psi$
    - **estimation of the violations ψ of origin/destionation information for P in the context given by ρ and κ**

3. Model Checking using OFMC

❖ Protocol description

- **Names: A, B (Alice, Bob), ...**
- **Keys: K, K^{-1} (inverse key),**
- **Encryption: $\{M\}_{K_A}$ (with A's public key), $\{|M|\}_{K_{AB}}$ (symmetric keys)**
- **Signing: $\{M_{K^{-1}}\}$**
- **Nonces: N_A**
- **Timestamps: T**
- **Messages: $\{M_1, M_2\}$**

❖ Example:

- $A \rightarrow B : \{A, T_A, K_{AB}\}_{K_B}$

3.1 Modeling Protocol Behavior

- **Example – Needham-Schroeder Public Key Protocol**

  1. $A \rightarrow B : \{N_A, A\}_{K_B}$

  2. $B \rightarrow A : \{N_A, N_B\}_{K_A}$

  3. $A \rightarrow B : \{N_B\}_{K_B}$

- **Proposed in 1970s, used for decades but wrong!**
- **People are pretty bad to understand all the interleavings.**

3.1 Modeling Protocol Behavior

- **Example – Needham-Schroeder Public Key Protocol**

  1. $A \rightarrow B : \{N_A, A\}_{K_B}$
  2. $B \rightarrow A : \{N_A, N_B\}_{K_A}$
  3. $A \rightarrow B : \{N_B\}_{K_B}$

- **Man-in-the-Middle Attack–two communications**

  1. $A \rightarrow I : \{N_A, A\}_{K_I}$
  2. $I(A) \rightarrow B : \{N_A, A\}_{K_B}$
  3. $B \rightarrow I(A) : \{N_A, N_B\}_{K_A}$
  4. $I(A) \rightarrow A : \{N_A, N_B\}_{K_A}$
  5. $A \rightarrow I(A) : \{N_B\}_{K_I}$
  6. $I(A) \rightarrow B : \{N_B\}_{K_B}$

- **B believes he is speaking with A!**

3.1 Modeling Protocol Behavior

❖ Man-in-the-Middle Attack–two communications

1. $A \rightarrow I : \{N_A, A\}_{K_I}$
2. $I(A) \rightarrow B : \{N_A, A\}_{K_B}$
3. $B \rightarrow I(A) : \{N_A, N_B\}_{K_A}$
4. $I(A) \rightarrow A : \{N_A, N_B\}_{K_A}$
5. $A \rightarrow I(A) : \{N_B\}_{K_I}$
6. $I(A) \rightarrow B : \{N_B\}_{K_B}$

❖ Corrected version: Needham-Schroeader-Lowe (1995)

1. $A \rightarrow I : \{N_A, A\}_{K_I}$
2. $I(A) \rightarrow B : \{N_A, A\}_{K_B}$
3. $B \rightarrow I(A) : \{N_A, N_B, B\}_{K_A}$ **– B should give his name**
4. $I(A) \rightarrow A : \{N_A, N_B, B\}_{K_A}$
5. A **aborts the protocol execution**

❖ Is the improved version now correct?

3.2 Protocol Analysis Using MC

❖ Model by Dolev & Yao

- a protocol as an algebraic system operated by the intruder
- perfect cryptography – all D_X private, decryption only with key, ...
- the intruder can read all traffic, modify, delete, create traffic, perform cryptographic operations, corrupt principals
- arbitrary number of principals
- protocol executions may be interleaved

❖ Modeling the Dolev-Yao Intruder

- M – set of messages
- $DY(M)$ – smallest set closed under generation G and analysis A rules

3.2 Protocol Analysis Using MC

❖ Modeling the Dolev & Yao Intruder

$$\frac{m \in M}{m \in DY(M)} G_{axiom} \qquad \frac{m_1 \in DY(M) \;\; m_2 \in DY(M)}{<m_1,m_2> \in DY(M)} G_{pair}$$

$$\frac{m_1 \in DY(M) \;\; m_2 \in DY(M)}{\{m_2\}_{m_1} \in DY(M)} G_{crypt} \qquad \frac{m_1 \in DY(M) \;\; m_2 \in DY(M)}{\{|m_2|\}_{m_1} \in DY(M)} G_{scrypt}$$

$$\frac{<m_1,m_2> \in DY(M)}{m_i \in DY(M)} A_{pair_i} \qquad \frac{\{|m_2|\}_{m_1} \in DY(M) \;\; m_1 \in DY(M)}{m_2 \in DY(M)} A_{scrypt}$$

$$\frac{\{m_2\}_{m_1} \in DY(M) \;\; m_1^{-1} \in DY(M)}{m_2 \in DY(M)} A_{crypt} \qquad \frac{\{m_2\}_{m_1}^{-1} \in DY(M) \;\; m_1 \in DY(M)}{m_2 \in DY(M)} A_{crypt}^{-1}$$

❖ Notes

- **generation (G), analysis (A)**
- $\{m_2\}_{m_1}$ **asymmetric encryption,** $\{|m_2|\}_{m_1}$ **symmetric encryption**
- G_{crypt} **– public key encryption,** G_{scrypt} **– symmetric encryption**
- A_{crypt} **– public key decryption,** A_{scrypt} **– symmetric decryption**

3.2 Protocol Analysis Using MC

❖ Modeling Protocol Behavior–A Trace-based Model

- **focus on communication traces**
- **a protocol describes a set of traces $\mathcal{M}$**
- **interleaving of runs of the protocol and messages from the attacker**

❖ Example: Needham-Schroeder

0.	$<>\in \mathcal{M}$	
1.	$t, A \rightarrow B : \{N_A, A\}_{K_B} \in \mathcal{M}$	**if** $t \in \mathcal{M}$
2.	$t, B \rightarrow A : \{N_A, N_B\}_{K_A} \in \mathcal{M}$	**if** $t \in \mathcal{M}$ **and** $A' \rightarrow B : \{N_A, A\}_{K_B} \in t$
3.	$t, A \rightarrow B : \{N_B\}_{K_B} \in \mathcal{M}$	**if** $t \in \mathcal{M}, A \rightarrow B : \{N_A, A\}_{K_B} \in \mathcal{M}$ **and** $B' \rightarrow A : \{N_A, N_B\}_{K_A} \in t$
4.	$t, Spy \rightarrow: M \in \mathcal{M}$	**if** $t \in \mathcal{M}$ **and** $M \in DY(IK_t)$

3.2 Protocol Analysis Using MC

❖ Modeling Property

- **a property also corresponds to a set of traces**

❖ Example: Authentication for A

- **If A used N_A to start a protocol run and with B received N_A back, then B sent N_A back.**

$$\textbf{A\_authenticates\_B(t)} \equiv \textbf{If } A \rightarrow B : \{N_A, A\}_{K_B} \in t \textbf{ and } B' \rightarrow A : \{N_A, N_B\}_{K_A} \in t \textbf{ then } B \rightarrow A : \{N_A, N_B\}_{K_A} \in t$$

$$\textbf{Spy\_Attacks\_A(t)} \equiv \neg \textbf{ A\_authenticates\_B(t)}$$

3.2 Protocol Analysis Using MC

Verification

- $\vdash \forall t. t \in \mathcal{M} \rightarrow$ **A_authenticates_B(t)**

Falsification

- $\mathcal{M} \vdash \exists t.$ **Spy_attacks_A(t)**

3.3 OFMC and AVISPA

- **see an example**
- **AVISPA project**

4. Conclusion

- **Analysis of security protocols**
  - **description using process calculi, traces etc**
  - **control flow analysis (static analysis)**
  - **model checking – using traces**
- **Tools**
  - LySa
  - **OFMC (On-the-fly Model Checker)**
  - **AVISPA – interface to OFMC**

5. References

❖ LYSA and Static Analysis

- C.Bodei, M.Buchholtz, P.Degano, F.Nielson, H.R.Nielson: Static Validation of Security Protocols, Journal of Computer Security, 2004.
- H.R.Nielson: Validation of Cryptographic Protocols using Static Analysis, lecture notes of VPSM 2006.

❖ OFMC/AVISPA

- David Basin: Model Checking Security Protocols using OFMC/AVISPA, lecture notes of VPSM 2006.
- AVISPA project