Automated Deduction and Security Protocols Analysis

Michaël Rusinowitch

Cassis/Nancy
Permanent members:
V. Cortier (CNRS)
S. Kremer (INRIA)
A. Imine (UNIV)
C. Ringeissen (INRIA)
M. Rusinowitch (INRIA)
M. Turuani (INRIA)
L. Vigneron (UNIV)
Team Cassis: automated validation tools for infinite state systems

- Automated Deduction
  - A rewriting approach to design decision procedures
  - security protocols: Avispa/Cl-Atse

- Safe Composition
  - security protocols
  - services: Avantssar/Orchestrator
  - collaborative applications: DesCal
Decision procedures for program verification

Derive decision procedures using:

1. Rewriting techniques
   - successful when formalizing data-structures

2. Combination techniques ex.
   \[ \text{store}(a, i, d)[j] \neq e \land f(i) = d \land i = j + k \]

Results: combining decision procedures for theories sharing

1. Counter arithmetics ex. \( \text{length}(\text{cons}(x, y)) = \text{length}(y) + 1 \)
2. The theory of Abelian Groups
Provably secure protocols

Issue 1  Modeling primitives accurately
For E-voting: e.g. blind signatures, re-encryption, trapdoor commitment, homomorphic encryption

Issue 2  Modeling and proving a wide range of security properties
For E-voting: e.g. privacy, coercion-resistance, verifiability

Issue 3  Transfer security from symbolic to computational models
Secure collaborative tools

Motivation

- Balance collaboration and access control
- Replication of data and access rights
- Decentralized access rights management

Contribution: Combining Operational Transformations with AC models

- Optimistic Access Control
- Collaborative tools: Shared Calendar for Mobiles

Remark: Google Docs & Calendar are centralized

Operational Transformation (OT)
Security Problems

Communication in Open Networks
- Alice and Bob exchange Messages
- An Intruder controls the communication channel

Security Objectives for: A sends message M to B
- **Confidentiality** (only A and B know M)
- Integrity of datas (M is not altered)
- Authenticity (B knows that A has sent M)
- Anonymity (A is not known ..)
- Non-repudiation (A cannot say he did not send M,...)

Solution: Security Protocols
- SSL: browsers, PGP: mail
- SET: E-commerce, Kerberos:remote login ....
Building Blocks for Security Protocols

Cryptographic Procedures: Encryption or signature of messages
- public keys \( \{ M \}_{K_B}, K_B^{-1} \Rightarrow M \)
- secret keys \( \{ M \}_K, K \Rightarrow M \)

(Pseudo-)Random Number Generators: to generate “Nonces”
- numbers used once e.g. for “Challenge-Response”

Protocols: recipe for exchanging messages
- Steps like: A sends B his name together with the message M. The pair \( \{ A, M \} \) is encrypted with B’s key.
  \[
  A \rightarrow B : \{ A, M \}_{K_B}
  \]
A secure protocol may be not so secure...

We need to verify secure communication schemes
Example: Replay attack

Alice → Bank: \{ transfer 1000 € to Charlie \}_{K_{Alice}^{-1}}

...
Example: Replay attack

Alice $\rightarrow$ Bank: \{ transfer 1000 € to Charlie \}$_{K^{-1}}^{Alice}$

Charlie $\rightarrow$ Bank: \{ transfer 1000 € to Charlie \}$_{K^{-1}}^{Alice}$

\ldots
Example: Needham-Schroeder authentication protocol

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

Meaning:

“I am Alice! and here is a nonce $N_A$."

“Here is your nonce. Since I could read it, I am Bob. And here is another nonce $N_B$."

“Thanks for $N_B$. As only Alice can read it, and here it is, I am Alice!”
Attack on Needham-Schroeder

1. \( A \rightarrow B : \{N_A, A\}_K_B \)
2. \( B \rightarrow A : \{N_A, N_B\}_K_A \)
3. \( A \rightarrow B : \{N_B\}_K_B \)

---

.. Bob croit qu’il parle avec Alice!
Attack on Needham-Schroeder

1. $A \rightarrow B : \{N_A, A\}_{K_B}$
2. $B \rightarrow A : \{N_A, N_B\}_{K_A}$
3. $A \rightarrow B : \{N_B\}_{K_B}$
Attack on Needham-Schroeder

1. $A \rightarrow B : \{N_A, A\}_{KB}$
2. $B \rightarrow A : \{N_A, N_B\}_{KA}$
3. $A \rightarrow B : \{N_B\}_{KB}$
Attack on Needham-Schroeder

1. $A \rightarrow B : \{N_A, A\}_K_B$
2. $B \rightarrow A : \{N_A, N_B\}_K_A$
3. $A \rightarrow B : \{N_B\}_K_B$

---

(loria)  Automated Deduction and Security Protocols
Attack on Needham-Schroeder

1. \( A \rightarrow B : \{N_A, A\}_{K_B} \)
2. \( B \rightarrow A : \{N_A, N_B\}_{K_A} \)
3. \( A \rightarrow B : \{N_B\}_{K_B} \)
Attack on Needham-Schroeder

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

\[ \text{NSPK #1} \]

\[ \text{NSPK #2} \]
Attack on Needham-Schroeder

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

..... Bob croit qu’il parle avec Alice!
Conséquence

\[ A(C) \rightarrow B: \{ A, N_B, \text{transférer 1000 } \text{€} \text{ d'}A \text{ à } C \}_B \]
Conséquence

A(C) → B: \{A, N_B, transférer 1000 € d’A à C \}_K_B

Correction par Lowe

1. A → B : \{N_A, A\}_K_B

2. B → A : \{N_A, N_B, B\}_K_A

3. A → B : \{N_B\}_K_B
Variante et attaque par confusion de types

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

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

3. $A \rightarrow B : \{N_B\}_{K_B}$
Variante et attaque par confusion de types

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

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

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

Si $C$ est un intrus ...

$C \rightarrow B: \{A, C\}_{K_B}$

$B \rightarrow A: \{C, N_B, B\}_{K_A}$

A pense que c’est le premier message de $C$

$A \rightarrow C: \{N_B, B, N_A, A\}_{K_C}$

C prend possession de $N_A, N_B$
RSA Protocol (with commuting encryptions)

1. $A \rightarrow B : \{L\}_{K_a}$

2. $B \rightarrow A : \{\{L\}_{K_a}\}_{K_b}$
RSA Protocol

3. \( A \rightarrow B : \{L\}_{K_b} \)
Attack on RSA Protocol

1. $A \rightarrow I(B) : \{L\}_{K_a}$

2. $I(B) \rightarrow A : \{\{L\}_{K_a}\}_{K_i}$
3. $A \rightarrow I(B) : \{L\}_{K_i}$

4.
Intruder Model (Dolev Yao)

He can

- spy, record, modify, reply
- masquerade by changing source address
- initiate parallel sessions
- (create type confusions)

He cannot

- decrypt without the key
- create cipher without both plaintext and encryption key
  
  e.g. $\{M\}_K, \{M'\}_K \not\models \{M \cdot M'\}_K$

black-box cryptography or perfect encryption hypothesis

... messages are first-order terms
Formal Analysis of Security Protocols

- Challenging as general problem is **undecidable**.
- Several **sources of infinity** in protocol analysis:
  - number of possible intruder messages.
  - message depth.
  - number of agents.
  - number of sessions or protocol steps.
  - number of possible values for nonces.

- Possible approaches:
  - **Falsification** identifies attack traces but does not guarantee correctness. ⇒ **constraint solving**
  - **Verification** proves correctness but difficult to automate (requires induction and often restrictions) ⇒ **(tree automata) approximation** or **resolution theorem proving**...
Summary of some known results

X means “bounded”
insecurity means “Intruder can derive Secret”

<table>
<thead>
<tr>
<th>msg size</th>
<th>nonces</th>
<th>sessions</th>
<th>insecurity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>undecided</td>
</tr>
<tr>
<td>X</td>
<td></td>
<td></td>
<td>[CC01]</td>
</tr>
<tr>
<td>X</td>
<td></td>
<td></td>
<td>undecided</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[D04]</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td></td>
<td>DEXP-complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[D04]</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td></td>
<td>NP-complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[AL01,RT01]</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>X</td>
<td>NP-complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[D04]</td>
</tr>
</tbody>
</table>
The AVISPA Tool
Intruder Rules

Decomposition:  Composition:

\[
\begin{align*}
\{a, b\} & \vdash a, b & a, b & \vdash \{a, b\} \\
K^{-1}, \{a\}_K & \vdash a & a, K & \vdash \{a\}_K \\
S, \{a\}_S & \vdash a & a, S & \vdash \{a\}_S \\
\end{align*}
\]

\[t \in \text{forge}(E) \iff E \vdash \ldots \vdash t, \ldots\]
Passive Intruder (only listening)

\[
\{ \text{secret} \}^{k_1}_{k_2}, \{ k_1, k_2 \}^{k_3}_{k_3}, k_3
\]
Passive Intruder (only listening)

\[\{\text{secret}\}_{k_1}^{k_2}, \{k_1, k_2\}_{k_3}, k_3 \vdash \{\text{secret}\}_{k_1}^{k_2}, k_1, k_2, k_3 \ldots\]
Passive Intruder (only listening)

\[
\{\text{secret}\}\{k_1\}_{k_2} , \{k_1, k_2\}_{k_3} , k_3
\]

\[
\vdash
\{\text{secret}\}\{k_1\}_{k_2} , \{k_1\}_{k_2} , k_3 \ldots
\]

\[
\vdash
\{\text{secret}\}\{k_1\}_{k_2} , \{k_1\}_{k_2} , k_3 \ldots
\]
Passive Intruder (only listening)

\[ \{ \text{secret} \} \{k_1\}_{k_2}, \{k_1, k_2\}_{k_3}, k_3 \]
\[ \vdash \]

\[ \{ \text{secret} \} \{k_1\}_{k_2}, k_1, k_2, k_3 \ldots \]
\[ \vdash \]

\[ \{ \text{secret} \} \{k_1\}_{k_2}, \{k_1\}_{k_2}, k_3 \ldots \]
\[ \vdash \]

\text{secret} \ldots
Intruder Constraint

Expression of type:

\[ E \triangleright t \]

with \( E \) a set of terms (e.g. some known messages) and \( t \) a term (message to forge).

A solution of an intruder constraint is a substitution \( \sigma \) such that

\[ \sigma(t) \in forge(\sigma(E)) \]

Generalize to systems of intruder constraints.
Encoding a Secrecy Problem using Constraints

1. \(A \rightarrow B: \{NA, A\}_{KB}\)
2. \(B \rightarrow A: \{NA, NB\}_{KA}\)
3. \(A \rightarrow B: \{NB\}_{KB}\)

gives system:

\[
\begin{align*}
&I_0, \{NA, A\}_{KB} \quad \triangleright \quad \{x, y\}_{KB} \\
&I_0, \{NA, A\}_{KB}, \{x, NB\}_{Ky} \quad \triangleright \quad \{NA, z\}_{KA} \\
&I_0, \{NA, A\}_{KB}, \{x, NB\}_{Ky}, \{z\}_{KB} \quad \triangleright \quad NB
\end{align*}
\]

with \(I_0\) initial intruder knowledge = \(\{KB, KA, A, B, I\}\)

\(NB\) can be obtained by Intruder iff the system is solvable
Upper bound

1. if a protocol $P$ is insecure for $n$ sessions then there is an attack where every sent message has size less than $|P| \times n$

2. $E \triangleright M$ can be checked in polynomial time: if the intruder can forge message $M$ from set of messages $E$, then he can do it in less than $|M| + |E|$ elementary operations

$\Rightarrow$ protocol insecurity for bounded number of sessions is in NP.
More precise models: algebraic properties

Bitwise XOR

\[
\begin{array}{cccccccc}
0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\
\oplus & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\
\hline \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
\end{array}
\]

1. \( A \rightarrow B \) : plaintext \( \oplus \) key

Subject to known-plaintext attack, since

plaintext \( \oplus \) ciphertext = key.
Extending Dolev-Yao Intruder by New Deduction Rules

\[ \{a, b\}_K \vdash \{a\}_K \]

for cipher block chaining attacks (ECB,CBC)

\[ a, b \vdash a \oplus b \]

where \( \oplus \) is associative, commutative, nilpotent with unit
Example of an open problem

**ac** theory: associative commutative operator +

**Composition rule:**

\[ a_1, a_2 \ldots \vdash_{ac} n_1 a_1 + n_2 a_2 + \ldots \]

where \( a_1, a_2, \ldots \) are terms and \( n_1, n_2, \ldots \) nonnegative integers

**How to solve systems:**

\[
\begin{align*}
\{ & l_0 \triangleright u_1 \\
& l_0, v_1 \triangleright u_2 \\
& \ldots \\
& l_0, v_1, \ldots, v_n \triangleright u_{n+1}
\}
\]
## Experiments with CL-ATSE (M. Turuani)

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Diagnosis</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASW - Abort</td>
<td>Secrecy flaw</td>
<td>0.03</td>
</tr>
<tr>
<td>DNSsec</td>
<td>Auth. flaw</td>
<td>0.98</td>
</tr>
<tr>
<td>EAP with IKEv2 Exp.</td>
<td>Safe</td>
<td>2.26</td>
</tr>
<tr>
<td>EKE Exp.</td>
<td>Auth. flaw</td>
<td>0.01</td>
</tr>
<tr>
<td>Fair Zhou-Gollmann</td>
<td>Auth. flaw</td>
<td>0.20</td>
</tr>
<tr>
<td>Fair Zhou-Gollmann (patch)</td>
<td>Safe</td>
<td>4.99</td>
</tr>
<tr>
<td>IKEv2 with MAC Exp.</td>
<td>Safe</td>
<td>7.38</td>
</tr>
<tr>
<td>IKEv2 with DS Exp.</td>
<td>Auth. flaw</td>
<td>0.03</td>
</tr>
<tr>
<td>Kerberos, cross-realm</td>
<td>Safe</td>
<td>0.32</td>
</tr>
<tr>
<td>Kerberos, forwardable ticket</td>
<td>Safe</td>
<td>0.14</td>
</tr>
<tr>
<td>Purpose Built Keys</td>
<td>Auth. flaw</td>
<td>0.01</td>
</tr>
<tr>
<td>Next Steps In Signaling</td>
<td>Safe</td>
<td>2.61</td>
</tr>
<tr>
<td>SET - Purchase Request</td>
<td>Secrecy flaw</td>
<td>0.17</td>
</tr>
<tr>
<td>SPEKE, with strong pwd. Exp.</td>
<td>Safe</td>
<td>0.04</td>
</tr>
<tr>
<td>SSH Exp.</td>
<td>Safe</td>
<td>1.22</td>
</tr>
<tr>
<td>TSIG</td>
<td>Safe</td>
<td>0.08</td>
</tr>
</tbody>
</table>
Perspectives: Protocol verification

- **Decision procedures in symbolic models**
  - For equivalence-based security properties (anonymity, e-voting)
  - For equational theories in the active case (e-voting cryptoprimitives)
  - Integrate security policies (e.g. revocation, delegation)

- **Modularity**
  - Design protocols in a modular, component-based way
Relating Formal and Cryptographic Approaches

<table>
<thead>
<tr>
<th></th>
<th>Formal approach</th>
<th>Cryptographic approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messages</td>
<td>terms</td>
<td>bitstrings</td>
</tr>
<tr>
<td>Encryption</td>
<td>idealized</td>
<td>algorithm</td>
</tr>
<tr>
<td>Adversary</td>
<td>idealized</td>
<td>any polynomial algorithm</td>
</tr>
<tr>
<td>Proof</td>
<td>automatic</td>
<td>by hand, tedious and error-prone</td>
</tr>
</tbody>
</table>

Link between the two approaches?
Web Service

**W3C**: software system designed to support interoperable machine-to-machine interaction over a network (internet)
- standardized data transmission via XML messages
- loose coupling among interacting services through standardized interfaces (WSDL)
- abstract communication layer provided by SOAP protocol and extensions
- support for composition (WS-BPEL, WS-CDL)

**Example**
Amazon WS for building business applications interacting with Amazon’s store. WSDL specification lists 40 operations on product database (ItemSearch, CartCreate, CartAdd ..). Can be used to implement clients that interact with Amazon WS (select best prices from Amazon and Fnac ..)
Web Services Composition

Why?

- Service Composability is one of the principles of Service Oriented Architectures
- A WS can rely on other WS to implement some of its internal computations

Security Policies

- Security Policies attached to WS constrain the composition (compatibility checking)
- Needs dynamic adaptation and explicit combination of applicable policies (negotiation)
Specification of the available services in ASLan v.1

(new) Service specified in ASLan v.1

The AVANTSSAR Validation Platform

Vulnerability: Policy: Tool input/output
P: Trust and Security
CS: Composed Service
CP: Composed Policy
S: Service

TS ORCHESTRATOR

orchestration/composition

Composed service/policy

CP

CS

TS Wrapper

validation problem

TS WRAPPER

secure

TS VALIDATOR

feedback

Vulnerability

: Tool input/output
P: Policy
S: Service
CP: Composed Policy
CS: Composed Service
TS: Trust and Security
Orchestration (Berardi et al. 2005)

\[ C \rightarrow G \rightarrow \ldots \]

\[ S_1 \quad S_2 \quad S_3 \ldots \]

"UDDI"

\[ M \rightarrow S_2 \rightarrow S_8 \rightarrow S_{14} \]

(loria) Automated Deduction and Security Protocols
Toy Example
Toy Example

Goal

\[(A+B)^2\]

\[\langle A,B \rangle\]

Goal

Adder

\[A+B\]

\[\langle A,B \rangle\]

Adder

Multiplier

\[A*B\]

\[\langle A,B \rangle\]
Toy Example with Security Policy: Mediator ~ Intruder

Client → Mediator
{<A,B>_K}

Mediator

{<A,B>_K1}

Adder → Mediator
{A+B}_K1

Mediator

Construct {<A,B>_K1}

{<A,B>_K1}

{A+B}_K1

Multiplier

Mediator

Construct {<A+B,A+B>_K2}

{<A+B,A+B>_K2}

Adder

Construct {(A+B)^2}_K2

Multiplier

{<A+B,A+B>_K2}

{(A+B)^2}_K2

Mediator

Construct {(A+B)^2}_K

{A+B}_K1

Multiplier → Mediator
{(A+B)^2}_K
<Body>
  <Order id=1>
    <beneficiary>Alice</>
    <account>Alice</>
    <trip>Paris</>
    <comment>super</>
  </Order>
</Body>
XML Injection

```xml
<Body>
  <Order id=1>
    <beneficiary>Alice</beneficiary>
    <account>Alice</account>
    <trip>Paris</trip>
    <comment> super </comment>
  </Order>
  <Order id=2>
    <beneficiary>Charlie</beneficiary>
    <account>Alice</account>
    <trip>Hawaii</trip>
    <comment></comment>
  </Order>
</Body>
```
Model for XML messages

nodes: free unary symbols $a()$, $b()$, ...

sons of a node: multiset

Possible deductions for attacker and honest agents:

- construct a node from a multiset of nodes
- extract the multiset of the sons of a node
- extract any node from a multiset of nodes
- construct a multiset from two already known multisets
Intruder deductions

For an XML node $a()$

\[
\begin{align*}
  & x \vdash a(x) \\
  & a(x) \vdash x
\end{align*}
\]

For multisets of sons with associative-commutative symbol $\cdot$

\[
\begin{align*}
  & x, y \vdash x \cdot y \\
  & x \cdot y \vdash x
\end{align*}
\]
Conclusion

Summary

- Apply automatic tool for verifying security protocols to compose web services under security constraints
- Intruder is the mediator and uses its capabilities to implement the goal service (e.g. adapting messages)
- Validation eliminates vulnerable compositions

Perspectives

- Generalize to protocols/services with loops and branching points, recursive tests
- Collaboration between several (domain specific) orchestrators
- Integrate access control policies
Références


Laurent Vigneron  Déduction automatique appliquée à l’analyse et la vérification de systèmes infinis. Habilitation à diriger des recherches, Université Nancy 2, novembre 2011