SATMC: SAT-based Model-Checking of Security Protocols

Roberto Carbone

Security&Trust, CIT_irst, Bruno Kessler Foundation, Trento, Italy

joint work with
Alessandro Armando, Luca Compagna, Luca Zanetti

Tool session of FOSAD 2013

Bertinoro, September 4, 2013
Outline

1. Introduction
2. LTL Model Checking for Security Protocol Analysis
3. Approach: SAT-based Model Checking of Security Protocols
4. Implementation
5. Demo
6. Usage and Results
7. Conclusions
Model checkers specifically tailored for security protocols have been remarkably successful in spotting flaws in protocols.

They rely on a number of simplifying assumptions:

**Dolev-Yao (DY) Channels:** controlled by an intruder, capable to overhear, divert, and fake messages.

**Honest Principals (HP):** required to react to messages of a specified form only.

**Security Goals (SG):** reachability properties.

Ok for simple protocols, but they prevent (or greatly complicate) the analysis of important real world protocols.
Model checkers specifically tailored for security protocols have been remarkably successful in spotting flaws in protocols.

They rely on a number of simplifying assumptions:

**Dolev-Yao (DY) Channels**: controlled by an intruder, capable to overhear, divert, and fake messages.

**Honest Principals (HP)**: required to react to messages of a specified form only.

**Security Goals (SG)**: reachability properties.

Ok for simple protocols, but they prevent (or greatly complicate) the analysis of important real world protocols.
Problems with the Common Assumptions

(DY) DY channels are not appropriate to model the behaviour of an attacker in

- **over-the-air protocols** (message interception unfeasible)
- **contract-signing protocols** (confidential, resilient channels)
- **browser-based protocols** (SSL/TLS channels)

(HP) Some protocols assume “non standard” behaviour of honest principals:

- **contract-signing protocols** (participants required to make progress)
- **browser-based protocols** (HTTP-redirect).

(SG) Some security goals cannot be (easily) expressed as reachability properties, e.g. fair exchange.
1 Approach to security protocol analysis based on model checking of LTL formulae.

2 The approach does not rely on (DY), (HP), and (SG).

3 Implementation in SATMC, a state of the art SAT-based Model Checker for security protocols.

4 Demo

5 Results: Effectiveness assessed against a number of real world protocols - **Severe flaws** found
Outline

1. Introduction

2. LTL Model Checking for Security Protocol Analysis

3. Approach: SAT-based Model Checking of Security Protocols

4. Implementation

5. Demo

6. Usage and Results

7. Conclusions
$M$: transition system modelling a superset of the behaviours of the honest agents and of the intruder.

$C_I$: LTL formula constraining the behaviours of the intruder.

$C_H$: LTL formula constraining the behaviours of honest principals.

$G$: LTL formula encoding the expected security property.

$LTL$ formula

\[
M \models (((C_I \land C_H) \Rightarrow G))
\]
LTL Model Checking for Security Protocol Analysis

Model $M$:

- $M$: transition system modelling a superset of the behaviours of the honest agents and of the intruder.
- $C_I$: LTL formula constraining the behaviours of the intruder.
- $C_H$: LTL formula constraining the behaviours of honest principals.
- $G$: LTL formula encoding the expected security property.

$$M \models ((C_I \land C_H) \Rightarrow G)$$
\( M \): transition system modelling a superset of the behaviours of the honest agents and of the intruder.

\( C_I \): LTL formula constraining the behaviours of the intruder.

\( C_H \): LTL formula constraining the behaviours of honest principals.

\( G \): LTL formula encoding the expected security property.
\[ M \models (((C_I \land C_H) \Rightarrow G)) \]

- \( M \): transition system modelling a superset of the behaviours of the honest agents and of the intruder.
- \( C_I \): LTL formula constraining the behaviours of the intruder.
- \( C_H \): LTL formula constraining the behaviours of honest principals.
- \( G \): LTL formula encoding the expected security property.
The Model

\[ M \models (C_i \land C_H) \Rightarrow G \]

Transition system associated with the concurrent execution of a number of sessions of the protocol.

- **States**: sets of facts, i.e. ground atomic formulae
- **Transitions**: rewrite rules define mappings between sets of facts.
### The Model: Facts

<table>
<thead>
<tr>
<th>Fact</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{state}_{\text{Role}}(j, a, es, s) )</td>
<td>Principal ( a ), playing role ( \text{Role} ), is ready to execute step ( j ) in session ( s ) of the protocol.</td>
</tr>
<tr>
<td>( \text{ak}(a, m) )</td>
<td>Principal ( a ) knows message ( m ).</td>
</tr>
<tr>
<td>( \text{sent}(rs, b, a, m, c) )</td>
<td>Principal ( rs ) has sent message ( m ) on channel ( c ) to principal ( a ) pretending to be principal ( b ).</td>
</tr>
<tr>
<td>( \text{rcvd}(a, b, m, c) )</td>
<td>Message ( m ) (supposedly sent by principal ( b )) has been received on channel ( c ) by principal ( a ).</td>
</tr>
</tbody>
</table>

Note: \( \text{ik}(m) \) abbreviates \( \text{ak}(i, m) \).

**Example (State):**

\[
\text{state}_{\text{Init}}(2, a, [ka, ka^{-1}, kb, na], 1) \cdot \text{sent}(a, a, i, \{\langle a, na\rangle\}_{ki}, c) \\
\cdot \text{state}_{\text{Resp}}(1, b, [kb, kb^{-1}, ka], 1) \cdot \text{ik}(ka) \cdot \text{ik}(kb)
\]
The Model: Rules for the Honest Agents

**Message Delivery**

\[
\text{sent}(RS, B, A, M, C) \xrightarrow{\text{receive}(A, B, RS, M, C)} \text{rcvd}(A, B, M, C) \cdot \text{ak}(A, M)
\]

**Message Processing**

\[
\text{rcvd}(A, B, M, C) \cdot \text{state}^{Role}(j, A, es, S) \xrightarrow{\text{send}_j(A, B, B_1, \ldots, S)} \text{sent}(A, A, B_1, M_1, C_1) \cdot \text{state}^{Role}(l, A, es', S)
\]
The Model: Rules for the Intruder

Interception

\[ \text{sent}(A, A, B, M, C) \xrightarrow{\text{intercept}(A, B, M, C)} \text{rcvd}(i, A, M, C) . \text{ik}(M) \]

Overhearing

\[ \text{sent}(A, A, B, M, C) \xrightarrow{\text{overhear}(A, B, M, C)} \text{sent}(A, A, B, M, C) . \text{rcvd}(i, A, M, C) . \text{ik}(M) \]

Faking

\[ \text{ik}(M) . \text{ik}(A) . \text{ik}(B) \xrightarrow{\text{fake}(A, B, M, C)} \text{sent}(i, A, B, M, C) . \text{ik}(M) . \text{ik}(A) . \text{ik}(B) \]
The Model: Inferential Capabilities of the Agents

\[\begin{align*}
\text{encrypt}(A, K, M) &\rightarrow ak(A, M). \text{ak}(A, K). \text{ak}(A, \{M\}_K) \\
\text{decrypt}_puk(A, K, M) &\rightarrow ak(A, \{M\}_K). \text{ak}(A, K^{-1}). \text{ak}(A, M) \\
\text{decrypt}_prk(A, K, M) &\rightarrow ak(A, \{M\}_{K^{-1}}). \text{ak}(A, K). \text{ak}(A, M) \\
\text{pairing}(A, M_1, M_2) &\rightarrow \text{ak}(A, M_1). \text{ak}(A, M_2). \text{ak}(A, \langle M_1, M_2 \rangle) \\
\text{decompose}(A, M_1, M_2) &\rightarrow \text{ak}(A, \langle M_1, M_2 \rangle). \text{ak}(A, M_1). \text{ak}(A, M_2)
\end{align*}\]
Constraining the Behaviour of the Intruder

\[ M \models (C_I \land C_H) \Rightarrow G \]

**Confidential Channel**

A *channel ch is confidential to principal p* iff its **output** is exclusively accessible to a given receiver p:

\[
\text{confidential}(ch, p) := G \forall (\text{rcvd}(A, B, M, ch) \Rightarrow A = p)
\]

**Resilient Channel**

Any message will be eventually delivered to the intended recipient.

\[
\text{resilient}(ch) := G \forall (\text{sent}(RS, A, B, M, Ch) \Rightarrow F \text{rcvd}(B, A, M, Ch))
\]

- Capital letters denote variables.
- \( \forall(\alpha) \) abbreviates the universal closure of \( \alpha \).
- Quantifiers are over finite domains (bounded analysis).
Constraining the Behaviour of Honest Principals

\[ M \models (C_I \land C_H) \implies G \]

Principal \( a \) should not indefinitely wait for an answer

\[ \Box \forall (\text{state}_R(j, a, \ldots) \implies \Box \neg \text{state}_R(j, a, \ldots)) \]

Received messages will be eventually processed by principal \( a \)

\[ \Box \forall (\text{rcvd}(a, P, M, C) \implies \Box \neg \text{rcvd}(a, P, M, C)) \]
Specifying Security Properties

\[ M \models (C_I \land C_H) \Rightarrow G \]

**Authentication**

*b authenticates a on m in session s* iff

\[
\text{authentication}(b, a, m, s) := 
G \forall (\text{state}_{rb}(\text{final\_step}, b, [a, \ldots, m, \ldots], s) \Rightarrow 
\exists O \text{state}_{ra}(\text{initial\_step}, a, [b, \ldots, m, \ldots], s))
\]

**Fair Exchange**

“\begin{quote}
A principal cannot obtain a valid contract without allowing the remaining principal to also obtain a valid contract.
\end{quote}

\[
G \forall (\text{ak}(a, contract) \Rightarrow F \text{ak}(b, contract))
\]
Outline

1. Introduction
2. LTL Model Checking for Security Protocol Analysis
3. Approach: SAT-based Model Checking of Security Protocols
4. Implementation
5. Demo
6. Usage and Results
7. Conclusions
SATMC reduces the security problem to propositional satisfiability problems (SAT).

Why SAT?
Dramatic speed-up of SAT solvers: problems with thousands of variables are now solved routinely in milliseconds.
SATMC: SAT-based Model Checking of Security Protocols

- SATMC reduces the security problem to propositional satisfiability problems (SAT).

- **Why SAT?**
  Dramatic speed-up of SAT solvers: problems with thousands of variables are now solved routinely in milliseconds.
Encoding to SAT

\[ \Phi^n = I(p_0) \land \bigwedge_{i=0}^{n-1} T_i(p_i, \lambda_i, p_{i+1}) \land GC(p_0, \ldots, p_n) \]

Additional **time-index** parameter to each **rule** \( \lambda \) or **fact** \( p \)

**Successful combination of**
- SAT-reduction techniques developed for AI-planning
- Bounded model-checking techniques for reactive systems
Encoding to SAT

\[ \Phi^n = I(p_0) \land \bigwedge_{i=0}^{n-1} T_i(p_i, \lambda_i, p_{i+1}) \land GC(p_0, \ldots, p_n) \]

**Additional time-index parameter to each rule \( \lambda \) or fact \( p \)**

**Successful combination of**

- SAT-reduction techniques developed for AI-planning
- Bounded model-checking techniques for reactive systems
Over-approximation of the reachable states

- **Idea:** Use knowledge about the initial state to simplify the $T_k$’s.
- **Approach:** Propagate information provided by the initial state for building an over-approximation of the forward search tree.
Over-approximation of the reachable states

- **Idea:** Use knowledge about the initial state to simplify the $T_k$'s.
- **Approach:** Propagate information provided by the initial state for building an over-approximation of the forward search tree.

### Linear Encoding

```
  I  T  T  T  T
```

### Graphplan-based encoding [2,3]

```
  I

  T_0  T_1  T_2  T_3
```

---

• **Pros**
  - leverages the speed-up of SAT solvers
  - **Expressivity:** LTL improves the scope of model checking for security protocols

• **Cons**
  - sometimes paid in terms of **efficiency**
Outline

1. Introduction
2. LTL Model Checking for Security Protocol Analysis
3. Approach: SAT-based Model Checking of Security Protocols
4. Implementation
5. Demo
6. Usage and Results
7. Conclusions
Implementation: Architecture

Security Problem (ASLan)

Graphplan-based Encoding

SAT Solving Interface

Model

Model2Attack

OK

Attack

K=0

\( \Phi^K \)

Max

Sover

MiniSat

ZChaff

...
1. Introduction

2. LTL Model Checking for Security Protocol Analysis

3. Approach: SAT-based Model Checking of Security Protocols

4. Implementation

5. Demo

6. Usage and Results

7. Conclusions
Demo: Toy Example
Outline

1. Introduction

2. LTL Model Checking for Security Protocol Analysis

3. Approach: SAT-based Model Checking of Security Protocols

4. Implementation

5. Demo

6. Usage and Results

7. Conclusions
SATMC is used in several research prototypes and industrial tools:

- Back-end of the AVISPA Tool and AVANTSSAR Platform and the back-end of the forthcoming SPaCioS Tool.
- Integrated in a SAP tool used to analyze SAP NetWeaver SAML Next Generation SSO.
- Used as an automated testcase generator in Tookan, a tool for analysing PKCS#11 security tokens.
Some Results

- **Contract Signing protocols**
  - Optimistic Fair Exchange Protocol by Asokan, Shoup, and Waidner
  - Flaw detected in a version of the protocol “patched” by Mitchell & Shmatikov


- **Strong authentication protocols**
  - user’s credentials + other proofs of identity
  - serious vulnerabilities in protocols for two-factor and two-channel authentication for web applications.
  - an attacker can carry out a security-sensitive operation by using only one of the two authentication factors.
Some Results

- **Contract Signing protocols**
  - Optimistic Fair Exchange Protocol by Asokan, Shoup, and Waidner
  - Flaw detected in a version of the protocol “patched” by Mitchell & Shmatikov


- **Strong authentication protocols**
  - user’s credentials + other proofs of identity
  - serious vulnerabilities in protocols for two-factor and two-channel authentication for web applications.
  - an attacker can carry out a security-sensitive operation by using only one of the two authentication factors.

Browser-based Security Protocols: Some Results

- Flaw detected in Google’s SAML-based SSO for Google Apps
- Authentication flaw in the most common use-case scenario of SAML 2.0 SSO Profile.
  (Errata by OASIS Security Services Technical Committee.)
- Cross-Site Scripting (XSS) vulnerabilities detected in:
  - SAML-based SSO for Google Apps
  - SimpleSAMLphp
  - Novell Access Manager v3.1
E90: RelayState sanitization

Security analysis of SAML implementations in [Sec2011] suggests that guidance is needed to advise implementers how to avoid enabling a class of attacks involving misuse of the RelayState feature supported by SAML bindings. The TC thanks the following for their identification of the problem, and their assistance in drafting this material:

- Alessandro Armando, University of Genova and Fondazione Bruno Kessler
- Roberto Carbone, Fondazione Bruno Kessler
- Luca Compagna, SAP
- Jorge Cuellar, Siemens
- Giancarlo Pellegrino, SAP
- Alessandro Sorniotti, IBM
- The EU Projects AVANTSSAR, SPaCloS, and SIAM

Add text to [SAMLBind] Section 3.1.1., before line 233:

New:

Some bindings that define a "RelayState" mechanism do not provide for end-to-end origin authentication or integrity protection of the RelayState value. Most such bindings are defined in conjunction with HTTP (and
E90: RelayState sanitization

Security analysis of SAML implementations in [Sec2011] suggests that guidance is needed to advise implementers how to avoid enabling a class of attacks involving misuse of the RelayState feature supported by SAML bindings. The TC thanks the following for their identification of the problem, and their assistance in drafting this material:

- Alessandro Armando, University of Genova and Fondazione Bruno Kessler

Outline

1. Introduction
2. LTL Model Checking for Security Protocol Analysis
3. Approach: SAT-based Model Checking of Security Protocols
4. Implementation
5. Demo
6. Usage and Results
7. Conclusions
We have presented a general framework for security protocols based on model checking of LTL formulae allowing for the specification of:

- assumptions on principals and channels
- complex security properties

that are normally not handled by state-of-the-art analysers.

SATMC: SAT-based Model Checking of Security Protocols

It works! Vulnerabilities detected on a number of important protocols:
ASW, SAML 2.0 SSO Profile, Google’s SAML-based SSO for Google Apps, Novell Access Manager, Strong Authentication protocols, ...
Conclusions

- We have presented a **general framework for security protocols** based on model checking of LTL formulae allowing for the specification of:
  - assumptions on principals and channels
  - complex security properties

that are normally not handled by state-of-the-art analysers.

- **SATMC**: SAT-based Model Checking of Security Protocols

- It works! Vulnerabilities detected on a number of important protocols:
  - ASW, SAML 2.0 SSO Profile, Google’s SAML-based SSO for Google Apps, Novell Access Manager, Strong Authentication protocols, ...
We have presented a general framework for security protocols based on model checking of LTL formulae allowing for the specification of:

- assumptions on principals and channels
- complex security properties

that are normally not handled by state-of-the-art analysers.

SATMC: SAT-based Model Checking of Security Protocols

It works! Vulnerabilities detected on a number of important protocols:
ASW, SAML 2.0 SSO Profile, Google’s SAML-based SSO for Google Apps, Novell Access Manager, Strong Authentication protocols, . . .
Thank you!

Acknowledgements:
This work was partially supported by

- **SPaCIoS**
  "Secure Provision and Consumption in the Internet of Services"

- **SIAM**
  "Automated Security Analysis of Identity and Access Management"