SECURITY THREAT IDENTIFICATION AND TESTING FOR SECURITY PROTOCOLS

Presented by Luca Compagna (SAP SE)
(joint work with Roberto Carbone, Annibale Panichella, Serena Ponta)
Context: Multi-Party Web Applications

Many modern web applications rely on TTPs to deliver services to their Users

• e.g., 27% of Alexa top 1000 uses Facebook SSO

Based on:

• protocols (interoperability)
• bilateral trust relationships

_TTPs are assumed to be trustworthy
But neither SP nor C are assumed so_
Challenges and Motivations

Several vulnerabilities reported in literature

Mainly implementation issues, but also design ones

Challenges include:
• highly configurable protocols, interpretation of the specifications
• internal requirements, total cost for development (TCD)
  • lack of (security) testing, but also
  • lack of tool support for developers
• ...

<table>
<thead>
<tr>
<th>Paper</th>
<th>Tech</th>
<th>Application(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sec.4 of [22]</td>
<td>FV</td>
<td>SPs implementing Google’s SAML SSO</td>
</tr>
<tr>
<td>Sec.5.2.1 of [30]</td>
<td>FV</td>
<td>SPs implementing OAuth 2.0 implicit flow-based Facebook SSO</td>
</tr>
<tr>
<td>Sec.IV.A.1 of [30]</td>
<td>BB</td>
<td>PayPal Payments Standard implementation in SPs using os-Commerce 2.3.1 or AbanteCart 1.0.4</td>
</tr>
<tr>
<td>Sec.V.A of [33]</td>
<td>WB</td>
<td>SPs implementing CaaS solutions 2Checkout, Chrono-Pay, PgiGate and Luottokunta (v1.2)</td>
</tr>
<tr>
<td>Sec.IV.A.2 of [30]</td>
<td>BB</td>
<td>PayPal Express Checkout implementation in SPs using OpenCart 1.5.3.1 or TomatoCart 1.1.7</td>
</tr>
<tr>
<td>Sec.4.2 of [34]</td>
<td>BB</td>
<td>SPs implementing OAuth 2.0 implicit flow-based Facebook SSO</td>
</tr>
<tr>
<td>Sec.6.2 of [23]</td>
<td>BB</td>
<td>developer.mozilla.com (SP) implementing BrowserID</td>
</tr>
<tr>
<td>Sec.V.C of [24]</td>
<td>FV</td>
<td>CitySearch.com (SP) using Facebook SSO (OAuth 2.0 Auth. Code Flow)</td>
</tr>
<tr>
<td>Sec.4 of [21]</td>
<td>FV</td>
<td>SPs implementing Google’s SAML SSO</td>
</tr>
</tbody>
</table>

Legend: FV: formal verification; BB: black-box; WB: white-box; M: manual inspection

[33] Sun, F., Xu, L., and Su, Z. Detecting logic vulnerabilities in e-commerce applications. NDSS 2014
Illustrative example
Developing and deploying SAML SSO

Assumption:
All HTTPS channels

Goal:
SP shall authenticate C

Goal:
resource shall be confidential

WHAT-IF:
IdP require signed SAML requests?

WHAT-IF:
SP field is not within the Assertion?

WHAT-IF:
SP does not store/check ID

SAML2 comes with many profiles, protocols, optional attributes, etc... + Internal requirements = several WHAT-IF
Illustrative example
Developing and deploying SAML SSO

source: few screen-shots of the SAP NetWeaver SAML Next Generation Single Sign On
Purpose: identify SAFE vs UNSAFE configurations in the WHAT-IF space
Illustrative example
Developing and deploying SAML SSO

<table>
<thead>
<tr>
<th>#</th>
<th>from</th>
<th>C-IdP</th>
<th>C-SP (1)</th>
<th>C-SP (2)</th>
<th>(1) = (2)</th>
<th>Authn Req</th>
<th>Assertions</th>
<th>Authn Resp</th>
<th>C checks on AuthnReq</th>
<th>SP checks ID</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Purpose:** identify SAFE vs UNSAFE configurations in the WHAT-IF space
Our solution

identify **SAFE vs UNSAFE** configurations in the **WHAT-IF** space

- Threat identification at design-time via model-checking
- Model-based testing

**rigorous, but viable** for an **industrial setting**

- accessibility / usability
- automation / integration
- cost-benefit ratio (TCO)
Our solution (cont.)

Sequence Diagram (informal model)

Front-end add-in for Enterprise Modelling Tools (SAP Power Designer)

Security Annotations (XMI files)

XMI Translator converts XMI files into Formal Models

Model Checker SATMC performs the formal analysis

Mutations mutate the formal models

Test Case Execution (model-based testing)
Scenario: SAML SSO

SAML 2.0 Web Browser SSO Profile:

- SAML-based SSO for Google Apps
- Novell Access Manager
- SimpleSAMLphp by UNINETT
Scenario: SAML SSO (demo)

SAML 2.0 Web Browser SSO Profile:
• SAML-based SSO for Google Apps
• Novell Access Manager
• SimpleSAMLphp by UNINETT

Vulnerabilities due to wrong design choices

Man-in-the-middle attack due to missing fields SP and ID in the assertion
Scenario: SAML SSO (demo)

SAML 2.0 Web Browser SSO Profile:
- SAML-based SSO for Google Apps
- Novell Access Manager
- SimpleSAMLphp by UNINETT

Vulnerabilities due to wrong design choices
Man-in-the-middle attack due to missing fields SP and ID in the assertion

Design vs. Implementation
XSS attack due to unrealistic assumption and missing input validation
Final remarks

Proof-of-concept READY
- prototype integrated within SAP Power Designer
- other use cases under scrutiny: e.g., mobile payment commercial solution

Potential end-users
- Architects and development teams integrating a core security protocol
- Security consultants analyzing a customer proprietary protocol (e-payment)
- Standardization bodies designing protocols and reference implementations

Industrial transfer (our experience)
- though lowered, the TCD is still not negligible
- consultancy mode works well, handing over the prototype not so well
THANK YOU

Contact: luca.compagna@sap.com