EXPERIENCE REPORT ON MODEL-BASED TESTING OF SECURITY COMPONENTS

Presented by Elizabeta Fourneret
EXPERIENCE REPORT ON MODEL-BASED TESTING OF SECURITY COMPONENTS

Frédéric Dadeau, Elizabeta Fourneret
Outline

• Introduction
• Our MBT approach
• Experimental results
• Conclusion and perspectives
Security Components

Security Components have two categories of test requirements:
- Functional Requirements
- Security Functional Requirements
Model-Based Testing

- **Modeling**
  - MBT model
  - Test Bench + Adaptation layer

- **Iterative process**
  - Manual test creation
  - Automatic test execution
  - Manual test execution

- **Test generation**
  - Tests
  - Coverage matrix

- **Deployment**
  - Component Under Test
  - Scripts
  - Scenarios

- **Req / Spec**
  - Manual test creation
Motivation

• Limitations of automated testing based on structural (for ex. requirement) coverage
  • test cases with limited size (steps)
  • difficulty to take into account the dynamics of the security functional requirements (must be hard-coded into the model)
  • possible issues with the test target’s reachability

• Our proposal: use temporal properties in TOCL and Test Purposes
  • How to express the test requirements easily?
  • How to characterize relevant tests?
Outline

• Introduction

• Our MBT approach

• Experimental results

• Conclusion and perspectives
Our MBT Approach

- Component Specification
- Security Functional Requirements
- MBT model
- Structural test selection
- Dynamic test selection
- Test repository
- PKCS#11 component*
- Certification
- Evaluation
- Coverage monitoring

*SoftHSM Virtual Cryptographic Token created by the group OPENDNSSEC
TOCL and TP test selection criteria

- **TOCL** and **TP** make possible to generate tests that exercise **corner cases**, relevant when testing security components.

- **TOCL** allows to express **temporal properties**, for instance of succession or precedence, contributing to the MBT process with:
  - Evaluation of the existing tests coverage
  - Verification of the model’s conformance to these properties
    - Simplifying the model debugging

- **TP** allow to express **procedures of tests** based on a verbose representation and using the experts experience and knowledge.
Outline

- Introduction
- Our MBT approach
- Experimental results
- Conclusion and perspectives
PKCS#11 and SoftHSM

- **PKCS#11** is an RSA standard that defines an interface called Cryptoki to promote interoperability and security of cryptographic tokens.

- **Scope:** 24 functions most commonly present in the tokens, such as session, token, key and user management functions, as well as cryptographic functions for signing messages and verifying signatures.

- To ensure the repeatability of the MBT process we chose SoftHSM - virtual cryptographic store largely used for exploring PKCS#11 without the necessity to posses an HSM (created by the group OPENDNSSEC).
Experimental results

Case study: PKCS#11

Component Under Test: SoftHSM

1\textsuperscript{st} experiment: evaluation of complementarity of test selection criteria to cover test requirements
2\textsuperscript{nd} experiment: evaluation of the error detection capabilities (robustness)
Experimental results

PKCS#11 set up metrics

<table>
<thead>
<tr>
<th>Test Requirement category</th>
<th>#FR</th>
<th>#SFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>general purpose</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>slot and token management</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>session management</td>
<td>32</td>
<td>9</td>
</tr>
<tr>
<td>object management</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>digesting</td>
<td>28</td>
<td>9</td>
</tr>
<tr>
<td>signing</td>
<td>32</td>
<td>10</td>
</tr>
<tr>
<td>verifying signatures</td>
<td>31</td>
<td>10</td>
</tr>
<tr>
<td>total</td>
<td>158</td>
<td>48</td>
</tr>
</tbody>
</table>

PKCS#11 model element

<table>
<thead>
<tr>
<th>PKCS#11 model element</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#classes</td>
<td>9</td>
</tr>
<tr>
<td>#enumerations</td>
<td>20</td>
</tr>
<tr>
<td>#enum. literals</td>
<td>123</td>
</tr>
<tr>
<td>#associations</td>
<td>17</td>
</tr>
<tr>
<td>#class attributes</td>
<td>34</td>
</tr>
<tr>
<td>#operations</td>
<td>24</td>
</tr>
<tr>
<td>#observations</td>
<td>1</td>
</tr>
<tr>
<td>#behaviors</td>
<td>206</td>
</tr>
<tr>
<td>#tcl properties</td>
<td>50</td>
</tr>
<tr>
<td>#test purposes</td>
<td>5</td>
</tr>
<tr>
<td>#LOC</td>
<td>1308</td>
</tr>
</tbody>
</table>

LOC: Lines of OCL constraints
Experimental results

PKCS#11 results metrics

<table>
<thead>
<tr>
<th>Test Selection Criterion</th>
<th>#Test targets</th>
<th>#Test cases</th>
<th>Cov. in %</th>
<th>FR</th>
<th>SFR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structural</td>
<td>206</td>
<td>184</td>
<td>100</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>TOCL</td>
<td>311</td>
<td>90</td>
<td>31</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>Test Purpose</td>
<td>24</td>
<td>24</td>
<td>9</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Manual</td>
<td>24</td>
<td>24</td>
<td>45</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Cost of applying the approach

~ 20 person / days

Fig. Distinct fault detection capabilities per coverage requirement
Conclusions of the study:

- **Relevance** of the TOCL and TP coverage criteria
  - “Produce tests that one may not easily think of”
  - augment test requirements coverage
- **TOCL and TP** increase fault detection capabilities
- **Usefulness** of coverage reports
  - show which part of the requirements are not covered by the tests
- **Cost-benefices:**
  - cost of applying TOCL and TP coverage criteria is very low
  - cost for *regression testing* (for ex. At the end of a sprint) is negligible
- **Use of the TOCL properties:** *model validation*
  - Use of the TOCL coverage measure to detect violations of the properties by the model
Outline

• Introduction

• Our MBT approach

• Experimental results

• Conclusion and perspectives
Conclusion

• We have experienced an MBT approach:
  • Combining static and dynamic test selection criteria
  • On a real-life security components

• Useful for:
  • Evaluating a test suite w.r.t. security requirement
  • Test selection, to augment a functional test suite
  • Increasing distinct fault-detection
Perspectives

• Test generation process
  • Online (fuzz) testing
  • Robustness criteria (based on TOCL automata coverage)

• Looking forward for other pilot projects to foreground our results.
Thanks for your attention!

Questions?