Debugging Support for Security Properties of Software Architectures

Kyungssoo Im, John D. McGregor
{kyungsi, johnmc}@cs.clemson.edu
School of Computing
Clemson University

Problem

• Software systems not meeting non-functional requirements is often the result of design level defects

• Difficult to analyze whether a modeled software architecture satisfies non-functional requirements, such as security
Background

• Software architecture is a high-level design describing the overall structure of a system
• Should satisfy all requirements: functional and non-functional
• Architecture description language (ADL) is commonly used to build a model
  – We will use Architecture Analysis and Design Language (AADL)
• Early design analyses/prediction are performed

Architecture Definition Process

[Diagram showing the process of architecture definition with labeled steps: Conceptual Architecture Modeling, Detailed Architecture Modeling, Architecture Model Simulation, and Rigorous architecture model.]
Cost of Defects in Architecture

Architectural Defects

- Much research in architecture level analyses
  - Tools and techniques
- Uniform goal is to analyze architecture before implementation to find and remove any defects at the design level
- We provide debugging support for security properties of software architectures
  - Search for the location of a fault in the architecture definition

Image from *CodeComplete* 2nd edition by Steve McConnell
Security as a Quality Attribute

- Security is considered by some to be a quality composed of
  - Availability: Software system’s percentage of uptime
  - Confidentiality: Only authorized users can access sensitive data
  - Integrity: Unauthorized data changes are not allowed

Approach

- Debugging a design for confidentiality defects relies on ability to measure confidentiality
- Authorization scheme is used by architects to assign values to the read authorization property of each element in the architecture
- Use of authorization scheme aids in controlling access to a resource and determining what access level is required to get to the resource
Authorization Levels in AADL

- Security requirement specified during requirements elicitation

    property set CUSE is
    readAuthorization: aadlinteger 1 .. 9 applies to (all);
    writeAuthorization: aadlinteger 1 .. 9 applies to (all);
    end CUSE;

- Ordinal scale represents access level required to invoke read/write operations in a particular component in software architecture

Scenarios

- Scenario-based approach is used
  – Identify elements where confidentiality attribute is greater than authorization level for a specific scenario

- In AADL, \textit{end-to-end flow} is used to describe a scenario
  – Logical flow of information from source to destination

- Determine if the sequence of activities are allowable by checking access levels specified in the path traveled by the flow
Example

flows
ETE3: *end to end flow*

\[
\text{T2.fs1} \rightarrow \text{conn2} \rightarrow \text{T1.fp1} \rightarrow \text{conn3} \rightarrow \text{T3.fsink} \\
\text{CUSE::readAuthorization} \Rightarrow 4;
\]

...

Our Tool

- Prototype tool developed to easily analyze if confidentiality and integrity quality attributes are satisfied
  - Built as AADL eclipse plug-in
- End-to-end flow in the model is traversed and identifies any access level violations
- Output shows detailed information to help locate places of violation
Output

Confidentiality Analysis Output

<table>
<thead>
<tr>
<th>Checking against ETE1 (Access Level 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidentiality Error: T1’s required level (3) != actor’s actual level (3)</td>
</tr>
<tr>
<td>Confidentiality Error: T2’s required level (4) != actor’s actual level (4)</td>
</tr>
<tr>
<td>OK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Checking against ETE2 (Access Level 7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidentiality Error: T2’s required level (7) != actor’s actual level (7)</td>
</tr>
<tr>
<td>OK</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Checking against ETE3 (Access Level 4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidentiality Error: T3’s required level (4) != actor’s actual level (4)</td>
</tr>
<tr>
<td>OK</td>
</tr>
</tbody>
</table>

Causes of Violation

- Once violations are identified
  - Need to fix or refine software architecture model to meet its quality attribute requirements, analogous to debugging in programming
- Causes of confidentiality or integrity violation
  - Scenario error
  - Access level is too high
  - Scenario actor’s level is too low
Role of Architect

- Based on the output from the tool
  - Examine location of violation in AADL model
  - Identify elements that do not match with the authorization level of the user of the flow
  - Modify individual elements, divide them or modify required authorization levels
  - Run same scenarios until desired quality attribute is satisfied by the AADL model

Added Benefits

- Defining and using the authorization scheme in AADL models gives added capabilities
  - Architecture slice: Shows only parts of the AADL model where sequence of activities taken for a scenario are shown
    - Reduces search space in finding defects
    - Compare slices of multiple failed scenarios to find similarity
Added Benefits

– Ability to make queries: Architects can perform queries to aid in understanding the possible travel paths in the architecture
  • Find all possible travel paths given a specific actor authorization level
  • Quickly identify unwanted paths traveled by a specific authorization level
– Handling multi-scale information
  • Uniform level of detail is not required for our approach
  • Varying levels of detail are accommodated

Summary

• Our approach provides capability to easily identify any shortcomings in the architecture description related to security quality attribute
  – Authorization scheme and tool-based approach
• Early determination and satisfaction of a security quality attribute helps in
  – Improving the quality of the software architecture
  – Positively influences the quality of the system built from the software architecture
Thank you

• Questions?