Generation of Scenario Graphs Using Model Checking

Somesh Jha (University of Wisconsin), Oleg Sheyner (CMU), Jeannette Wing (CMU)
Example of Attack Graph Developed by a Professional Red Team

Sandia Red Team “White Board” attack graph from DARPA CC20008 Information battle space preparation experiment

Drawn By Hand
Definitions

• Given
  – a finite state model $M$
  – a correctness property $\Phi$

• An failure scenario is an execution of $M$ that violates $\Phi$.

• An scenario graph is a set of failure scenarios of $M$. 
Properties of Scenario Graphs

• **Exhaustive**
  – All possible failure scenarios are represented in $G$.

• **Succinct**
  – Only relevant states are contained in $G$.
  – Only relevant transitions are contained in $G$. 
Problem Statement

• **Problem:** Generating scenario graphs by hand is tedious, error-prone, and impractical for large systems.

• **Our Goal:** Automate the generation and analysis of scenario graphs.

  – **Generation**
    - Must be fast and completely automatic
    - Must handle large, realistic examples
    - Should guarantee properties of scenario graphs

  – **Analysis**
    - Enables tool-aided post-generation analysis
Overview of Our Method

Phase 1

System Model → Generator → Scenario Graph

Phase 2

Annotations

Query: What system transitions lead to failure?

Minimization Analyzer → Scenario Subgraph

Reliability Analyzer → Probabilistic Scenario Graph

Query: What is the likelihood of failure?

... → Cost Analyzer
Symbolic Scenario Graph Generation

• **Inputs**
  - \( S, S_0 \subseteq S, R \subseteq S \times S \)
  - \( \Phi = AG (\neg \text{unsafe}) \) (a safety property)

• **Output**
  Scenario graph \( G = (S_{\text{unsafe}}, S_0^\Phi, R^\Phi) \)

• **Algorithm**
  1. \( S_{\text{unsafe}} = \text{modelCheck}(S, S_0, R, \Phi) \)
     (* Use an iterative algorithm derived from the fixpoint characterization of \( AG \) operator. *)
  2. \( S_0^\Phi = S_0 \cap S_{\text{unsafe}} \)
  3. \( R^\Phi = R \cap (S_{\text{unsafe}} \times S_{\text{unsafe}}) \)
Explicit-State Scenario Graph Generation

• Based on Automata-Theoretic Model Checking
  
  – Interpret both model $M$ and correctness property $\Phi$ as Buchi automata.
  
  – $M$ and $\Phi$ induce languages $L(M)$, $L(\Phi)$.
  
  – $L(M) \setminus L(\Phi) = \text{executions of } M \text{ that violate } \Phi$.
  
  – Construct $M \cap \sim \Phi$ by computing intersection of Buchi automata.

• $\Phi$ can be any LTL property.
Explicit-State Algorithm Illustrated

Model $M$

LTL Property $\neg F c$

Generation of Scenario Graphs Using Model Checking

HCES 05/01/2003
Explicit-State Algorithm (Cont.)

Find strongly connected components (SCCs) (R. Tarjan ’72)

Collect SCCs with acceptance states

Add paths from initial states
Performance

Linear Regression $R^2 = 0.9967$
## State Hashing

<table>
<thead>
<tr>
<th>Method</th>
<th>Full State Size</th>
<th>Hashcompact</th>
<th>Traceback</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance (Amortized)</td>
<td>O(E)</td>
<td>O(E)</td>
<td>O(E)O(depth)</td>
</tr>
<tr>
<td>Completeness</td>
<td>Complete Coverage</td>
<td>Partial Coverage</td>
<td>Complete Coverage</td>
</tr>
<tr>
<td>Memory Overhead per State</td>
<td>Full State Size</td>
<td>8 bytes</td>
<td>14 bytes</td>
</tr>
</tbody>
</table>
Example Attack Graph

Security property (LTL):

\( G (\text{intruder.privilege(host)} < \text{root}) \)

![Diagram of an attack graph with nodes for local buffer overflow (CVE-2002-0004), IIS buffer overflow (CAN-2002-0364), Squid portscan (CVE-2001-1030), LICQ remote-to-user (CVE-2001-0439), and the final state labeled "Done!".]
Application: Attack Graphs

- System and Goal Specification
- Model Builder
- Attack Graph Generators
- Attack Graph Analyzers
- Graphical User Interface
- Outpost Server
- SQL database
- MITRE
- Host Configuration Data
- Network Configuration Data
- Outpost Clients

Generation of Scenario Graphs Using Model Checking

HCES 05/01/2003