Combinatorial Methods for Dynamic Gray-Box SQL Injection Testing

Bernhard Garn, Jovan Zivanovic, Manuel Leithner & Dimitris E. Simos

Problem & Motivation

SQL injection remains one of the most popular classes of security flaws in web applications. While modern frameworks prevent SQLI in most situations, security testing is still required. Popular tools primarily use hardcoded lists of exploit vectors.

SQLInjector+
- Grammar-based exploit vector generation
- Selected & refined based on context in query
- Evaluated against popular security testing tools

Approach
- Context-specific grammars
  - Optimized for specific location in query
  - Adapted at runtime, avoids filtered characters
  - E.g. WHERE, LIMIT, ORDER, ...
- Combinatorial coverage guarantees
  - Guaranteed coverage and diversity
  - Small, efficient test sets
  - Strength chosen by user
- DBMS proxy
  - Intercepts queries
  - Identifies injection context
  - Supports oracle decision

Evaluation
- Tools: sqlmap, w3af, wapiti, FuzzDB, SQLInjector (predecessor)
- Training targets: WAVSEP, DVWA, Web4Pentester, MCIR-SQLOL
- Real-world web apps: WebChess, geccBBite, zeusCart, OpenSchool, ModSecurity (with synthetic web app)

Results
Detection rate (DR): Ratio of successfully exploited endpoints over all endpoints.

<table>
<thead>
<tr>
<th>Tool</th>
<th>DCRI</th>
<th>DCRI (T)</th>
<th>DCRI (S)</th>
<th>DCRI (T)</th>
<th>DCRI (S)</th>
<th>DCRI (T)</th>
<th>DCRI (S)</th>
<th>DCRI (T)</th>
<th>DCRI (S)</th>
<th>DCRI (T)</th>
<th>DCRI (S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>sqlmap</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w3af</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wapiti</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FuzzDB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQLInjector</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>

Proven efficacy

Our prototype implementation SQLInjector+ finds previously undetected flaws and leads the pack in a comparison with the state of the art (sqlmap, w3af, wapiti...).

Figure 1: Detection Rate across all SUTs; higher is better.

Conclusion

- Context-specific, gray-box combinatorial security testing
- Geared towards SQL injections, theoretically applicable to other classes
- Adjust test set size based on resource limits and preference
- Finds new vulnerabilities
- Identifies more flaws than state of the art
- Offers various string tainting & escape techniques


This research was further funded by the Austrian Research Promotion Agency (FFG): Grant no. 865248 (SECuring Web technologies with combinatorial Interaction Testing - SecWIT).