Internal and External Metrics for Predicting Attack-prone Components

Michael Gegick and Laurie Williams
North Carolina State University
7 April 2008
Where Should Security Efforts Begin?

(Reability context)
Fault-prone component
Likely to contain faults

Failure-prone component
Likely to fail

(Security context)
Vulnerability-prone component
Likely to contain vulnerabilities

Attack-prone component
Likely to be exploited

Fault- and vulnerability-prone
- Pre-execution context
- Some faults remain latent.
- Vulnerabilities can have a wide range of severity and likelihood of exploitation.

Failure- and attack-prone
- Execution context
- Execution of a fault is a failure.
  - Usage
- Exploitation of a vulnerability is an attack.
  - Ease of attack and value of asset (risk)
• **Goal** - identify *where* vulnerabilities most likely exist in a software system so fortification efforts can focus on those problem areas first.

• **Research objective** – create/validate statistical models that identify good and early predictors of security problems.

• **Candidate predictors**
  – Churn
  – Size (SLOC)
  – FlexeLint static analysis tool alerts (audited and un-audited)
    • All alerts
    • Null pointers
    • Memory leaks
    • Buffer overflows
  – Non-security failures (general reliability problems)

• **Methodology** - model values of the predictors and counts of security-based failure reports for a given component in the software system.

• *Not* identify exploits or qualify the vulnerabilities.
Case Study

- Commercial telecommunications software system.
- 38 components
  - 13 components left out $\rightarrow$ 25 components in analysis
  - Each component consists of multiple files
- 1.2 million lines of C/C++ source code (in the 25 components)
- Deployed to the field for two years
- 52 failure reports were classified as security-based problems
  - Vendor’s security engineer verified our report
Attack-prone Components

- **Pre-release attack-prone components (10)**
  - Pre-release robustness testing at system level

- **Post-release attack-prone components (4)**
  - Customer-reported
    - “attacks” – vulnerabilities that could have been exploited
      » No attacks reported

- Attack-prone (not vulnerability-prone)
  - Vulnerabilities were found during system execution

- All post-release attack-prone components are also pre-release attack-prone
## Correlations

<table>
<thead>
<tr>
<th>Metric</th>
<th>Security failure count</th>
<th>Spearman rank correlation (p-value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FlexeLint alerts</td>
<td>Sum pre- and post-release</td>
<td>0.39 (.06)</td>
</tr>
<tr>
<td>Churn</td>
<td>Pre, post- or both</td>
<td>No correlation</td>
</tr>
<tr>
<td>SLOC</td>
<td>Post-release</td>
<td>0.43 (0.03)</td>
</tr>
<tr>
<td>Sum pre- and post-release non-security failure count</td>
<td>Sum pre- and post-release</td>
<td>0.82 (&lt; .0001)</td>
</tr>
</tbody>
</table>
## Classification and Regression Tree Analysis (CART)

<table>
<thead>
<tr>
<th>32% correctly classified as not AP</th>
<th>20% correctly classified as not AP</th>
<th>40% correctly classified as AP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>churn &lt; 3861</strong></td>
<td><strong>Total alert density (audited) &lt; 0.19</strong></td>
<td><strong>Total alert density (audited) ≥ 0.19</strong></td>
</tr>
<tr>
<td><strong>Lower partition (first split)</strong></td>
<td><strong>Upper partition (second split)</strong></td>
<td><strong>Upper partition (second split)</strong></td>
</tr>
</tbody>
</table>

**Lower partition (second split)**

**Second split**

**Upper partition (second split)**
## Attack-prone Prediction Results from CART

<table>
<thead>
<tr>
<th>Metric</th>
<th>Type I</th>
<th>Type II</th>
<th>R²</th>
<th>Cross-validated R²</th>
<th>ROC</th>
</tr>
</thead>
<tbody>
<tr>
<td>alerts</td>
<td>7 (28%)</td>
<td>0%</td>
<td>31.5%</td>
<td>19.4%</td>
<td>76.7%</td>
</tr>
<tr>
<td>churn</td>
<td>7 (28%)</td>
<td>0%</td>
<td>32%</td>
<td>30%</td>
<td>77%</td>
</tr>
<tr>
<td>SLOC</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>alerts, churn, SLOC</td>
<td>2 (8%)</td>
<td>0%</td>
<td>68%</td>
<td>61%</td>
<td>93%</td>
</tr>
<tr>
<td>total pre-release failure count</td>
<td>2 (8%)</td>
<td>0%</td>
<td>68%</td>
<td>64%</td>
<td>93%</td>
</tr>
</tbody>
</table>
Non-security and Security Failure Counts

All post-release attack-prone components are also pre-release attack-prone components.
Pre-release non-security failures are good predictors of pre- and post-release security failures (in our setting).

- Negative binomial distribution
  - Standard error = 0.56
  - p<.0001
  - Value/DF = 0.92
Limitations

• Small sample size – 25 components

• Moderate R² values

• Only one data set

• Only one static analysis tool
  – Not representative of all static analysis tools.

• Testing effort not necessarily equivalent on all components
The Coupling Effect

- Coupling effect – “simple” problems found by FlexeLint are **coupled** to more complex problems in design and operation.
  - E.g. - buffer overflow (simple) in same file as an access control issue.
    - Developer does not understand buffer overflows (a potential security problem) which could indicate that they do not understand the encryption requirements for an authentication mechanism.
    - Customer requirements are unclear $\rightarrow$ design is ambiguous$^1$ $\rightarrow$ developers make guesses about the ambiguous designs.
  - Failure reports
    - 60% - coding bugs (**hopefully found by static analysis tools**)
    - 40% - design flaws and operational vulnerabilities
    - The “simple” 60% can predict the “complex” 40%

Summary

• Components with high code churn and FlexeLint alerts are attack-prone.

• Components with many non-security failures are attack-prone.

• Reliability testers can find security vulnerabilities.
Looking for industrial partners!

Thank you!

mcgegick@ncsu.edu
williams@csc.ncsu.edu