Automated Thrash Testing

By Andre Gironda

OWASP September 2007
Bio

- Andre Gironda
- Chicago / OWASP
- Second best security blog commenter in all of Kazakhstan
Advice from former managers

• Remember these words (quickly forgot)
• Ask the right questions
• In Infosec, terminology is everything
• Listening skills are critical (hear that? Good)
Current situation

- RIA frameworks
- Marketing vs. security
- Customer service
- SaaS, SOA, Web 2.0\wned
- Ajax security models
  - Application logic accessible on the client
Outline of this talk

• OWASP: Problem to solve
• Model vs. measure
• Models to measure testing tools
• A brief interlude into the dev & QA worlds
• How to report findings and fix them
• Prediction of future
**OWASP: Problem to solve**

- Automated Thrash Testing
  - Thrash vs. fuzz
  - Terminology is important
  - Meaningless words / acronyms must evolve

<table>
<thead>
<tr>
<th>Narrowband</th>
<th>Boundary value analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wideband</td>
<td>Fault-injection</td>
</tr>
<tr>
<td>Broadband</td>
<td>Fuzz testing</td>
</tr>
<tr>
<td>DWDM</td>
<td>Thrash testing</td>
</tr>
</tbody>
</table>
Maturity models

- The language of business
- SSE-CICISMTMM
  - Systems Security Engineering
  - Continuous Integration
  - Capability MM
  - ISM3
  - Testing MM
  - Integrated!
- Model vs. measure (Jaquith)
OWASP Software Security Tool Maturity Model

- It’s about tools
- OSSTMM
  - For pen-testers
  - OSSTMM v3
  - Book: Annotated OSSTMM
- You have to wait until the end of the talk
The other side of the house

• Development testing & inspection
  – Types of testing
Intake testing: Keep the bar green

- Developer freebies in their IDE/SCM (warn2err)
- Static source code analysis
- Coding standards
- Static binary/bytecode analysis

- Continuous-testing IDE with decision coverage

- Unit testing, “Never in the field of software development was so much owed by so many to so few lines of code.” – Martin Fowler pretending to be Winston Churchill
Smoke testing: Build every day

- Timed releases – daily builds
  - ThoughtWorks Buildix boot CD
    - Subversion, Trac, CruiseControl, User manager
    - Atlassian JIRA/Confluence, FishEye, Bamboo
    - Luntbuild, ViewVC, Hudson
- Component tests (DB, mock/stub)
- System tests
- Metrics
Inspection! Review the code

- Major builds – securecoding (SC-L)
- Fagan inspection
- Peer review
  - Author
  - Reviewer
  - Moderator
- Continuous inspection
Release of a webapp

- Model-checking
- Smart fuzz testing
- Concolic unit testing

- Two reasons to do this (Gadi Evron)
  - Fuzz before release
  - Fuzz before purchase
System integration test

- Test the server in working environment
- Components, components, components
- Script-driven, domain-specific languages
  - Protocol drivers, proxy fuzzers
- Data-driven test frameworks
Functional testing

• Test the client
• Simulate or drive browsers and plug-ins
  – Application drivers
• Repeatable tests
• Capture/playback test frameworks
Best of all worlds

- Continuous dev/QA/security integration

<table>
<thead>
<tr>
<th>Developer</th>
<th>Intake &amp; smoke</th>
<th>Build server -</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Code review</td>
<td>Ant tasks</td>
</tr>
<tr>
<td>Software quality engineer</td>
<td>Functional</td>
<td>Multi-driver -</td>
</tr>
<tr>
<td></td>
<td>Regression</td>
<td>WebDriver</td>
</tr>
<tr>
<td>Security professional</td>
<td>Acceptance</td>
<td>Web application</td>
</tr>
<tr>
<td></td>
<td>Maintenance</td>
<td>vuln scanners?</td>
</tr>
</tbody>
</table>
What to include in findings

• Which cheat-sheet / taxonomy used?
  – Input values + results format in a table
• Experienced-based (exploratory) testing?
• Does this defect remind you of an old one (VulnDB)?
• Scoring?
Back to threat-models

- Re-design! (back to the drawing board)

<table>
<thead>
<tr>
<th>Attack-trees</th>
<th>MITRE CAPEC</th>
<th>WASC TC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seven pernicious kingdoms</td>
<td>CWE</td>
<td>OWASP T10</td>
</tr>
<tr>
<td>STRIDE</td>
<td>X.805</td>
<td>Trike</td>
</tr>
</tbody>
</table>
Back to development

• Continuous-prevention development
Bonus: Assert others by looking for defect’s fix
Multiple Tool Evaluation Criteria

- Basic criteria
- FN vs. FP
- Non-exploitables?
- TP’s vs. testing ground
  - OWASP SiteGenerator
  - Stanford SecuriBench
Single Tool Evaluation Criteria

• Advanced criteria
• NIST SAMATE Evaluation Criteria
  – Bug categories (CWE, OWASP, WASC, PCI)
  – Levels of defense

\[
\frac{100 \cdot t}{t + p + n}
\]

• 100 * TP / TP + FP + FN (Brian Chess)
The Future

• Hybrid tools and hybrid people?
• Logical vs. semantic (Curphey’s flaws vs. bugs)

OWASP DC on RIA: http://www.owasp.org/index.php/RIA_Security_Smackdown


ISECOM’s OSSTMM: http://www.isecom.org


Promoting Warnings to Errors: http://safari5 bvdep.com/9780596510237/enabling_useful_warnings_disabling_useless_ones_and_promoti


Refs (cont’d)

SecureCoding Mailing-list: http://www.securecoding.org/list/
Atlassian (formerly Cenqua) Crucible: http://www.atlassian.com/software/crucible/
Concolic testing: http://osl.cs.uiuc.edu/~ksen/cute/
Fuzzing in the corporate world, Gadi Evron:
http://events.ccc.de/congress/2006/Fahrplan/events/1758.en.html
GPath with XmlParser and NekoHTML:
Twill: http://twill.idyll.org MaxQ: http://maxq.tigris.org
OpenQA Selenium, Watir: http://openga.org TestGen4Web:
Brian Chess & Katrina Tsipenyuk:
http://securitymetrics.org/content/attach/Welcome_blogentry_010806_1/software_che ss.ppt