Toward Black-box Detection of Logic Flaws in Web Applications

Giancarlo Pellegrino
gpellegrino@deeds.informatik.tu-darmstadt.de

Davide Balzarotti
davide.balzarotti@eurecom.fr

San Diego, 25/02/2014
Agenda

- Problem
- Approach
  - Model Inference
  - Behavioral Patterns Extraction
  - Attack Pattern-based Test Case Generation
  - Test Execution and Oracle
- Evaluation
- Conclusion
Logic Flaws

- Also known as design flaws/errors, business/application logic errors/flaws
- Lack a formal definition
  - CWE-ID 840: Business logic errors are “weaknesses […] that commonly allow attackers to manipulate the business logic of an application”
- Mainly caused by insufficient validation of the application workflow and data flow
- Can exhibit patterns, e.g.
  - Improper authentication/authorization
<table>
<thead>
<tr>
<th>Source code</th>
<th>Explicit Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
White-box testing \cite{BalzarottiCCS07, FelmetsgerUSENIX10, ...} 
- Source code of WA may not be available $\rightarrow$ White-box not applicable!
## Problem

### White-box testing  
[BalzarottiCCS07, FelmetsgerUSENIX10, ...]
- Source code of WA may not be available → White-box not applicable!

### Design verification  
[LoweCSF97, ArmandoCSF07, ...]
- Specification of WA may not be available → DV not applicable!
## Problem

<table>
<thead>
<tr>
<th>Source code</th>
<th>Explicit Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Black-box</td>
</tr>
<tr>
<td></td>
<td>White-box</td>
</tr>
<tr>
<td></td>
<td>Design verification</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Black-box</td>
</tr>
<tr>
<td></td>
<td>Design verification</td>
</tr>
</tbody>
</table>

- **White-box testing**  [BalzarottiCCS07, FelmetsgerUSENIX10, ...]
  - Source code of WA may not be available → White-box not applicable!

- **Design verification**  [LoweCSF97, ArmandoCSF07, ...]
  - Specification of WA may not be available → DV not applicable!

- **Black-box testing, e.g., web scanners**  [DoupèDIMVA10, WangS&P11, WangS&P12]
  - Cannot automatically detect logic flaws

  ➔ *Testing for logic flaws is done manually*
Our Approach
Overview

1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Resource Abstraction

Resource Clustering

2) Behavioral Patterns

3) Test Cases Generation

3) Test Cases Generation

4) Test Cases Execution

Execution

Oracle

Verdict:
Flaw found in test 1 and 2
Model Inference

1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

2) Behavioral Patterns

3) Test Cases Generation

4) Test Cases Execution

Resource Abstraction

Resource Clustering

Data flow Patterns

Workflow Patterns

PChain 1

PChain 2

Test Cases

Execution

Verdict: Flaw found in test 1 and 2
Behavioral Patterns Extraction

1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

2) Behavioral Patterns

Resource Abstraction

Data flow Patterns

Workflow Patterns

3) Test Cases Generation

4) Test Cases Execution

Oracle

Execution

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

Verdict: Flaw found in test 1 and 2
Workflow Patterns

Traces:

\[ \pi_1 = \langle a, b, a, c, d, e, f, e \rangle \]
\[ \pi_2 = \langle a, c, d, e, f, e \rangle \]
Workflow Patterns

Traces:
\[ \pi_1 = \langle a, b, a, c, d, e, f, e \rangle \]
\[ \pi_2 = \langle a, c, d, e, f, e \rangle \]

Model:

TrWP : Trace Waypoints
Rp : Repeatable Operations
Data flow Patterns

Trace 1:

   <HTML>
   <a href="/view.php?tok=8AFFB0">[…]

   <HTML>
   <a href="/add.php?tok=8AFFB0">[…]

   <HTML>
   <a href="/checkout">[…]

Trace 2:

   <HTML>
   <a href="/add.php?tok=DDA124">[…]

   <HTML>
   <a href="/checkout">[…]
Test Case Generation

1) Model Inference

```
74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89
```

2) Behavioral Patterns

```
74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89
```

3) Test Cases Generation

```
Test Cases
```

4) Test Cases Execution

```
Execution
```

Verdict: Flaw found in test 1 and 2
Attack Pattern-based Test Case Generation

Multiple Execution of Repeatable Singletons

Breaking Multi-Step Operations

Breaking Server-Generated Propagation Chains

Waypoints Detour
Attack Pattern-based Test Case Generation
Test Case Execution and Oracle

1) Model Inference

74.125.230.240 > 192.168.1.89
192.168.1.89 > 74.125.230.240
74.125.230.240 > 192.168.1.89

2) Behavioral Patterns

3) Test Cases Generation

4) Test Cases Execution

Verdict: Flaw found in test 1 and 2
1) Model Inference

\[74.125.230.240 \rightarrow 192.168.1.89\]
\[192.168.1.89 \rightarrow 74.125.230.240\]
\[74.125.230.240 \rightarrow 192.168.1.89\]

2) Behavioral Patterns

\[\text{Resource Abstraction} \rightarrow \text{Resource Clustering}\]

3) Test Cases Generation

\[\text{Security Property:}\]
\[ord_{\text{placed}} \land on\text{Store}(S) \implies\]
\[0(paid(U, I) \land to\text{Store}(S) \land\]
\[0(ack(U, I) \land on\text{Store}(S)))\]

4) Test Cases Execution

\[\text{Execution:}\]
\[74.125.230.240 \rightarrow 192.168.1.89\]
\[192.168.1.89 \rightarrow 74.125.230.240\]
\[74.125.230.240 \rightarrow 192.168.1.89\]

\[\text{Verdict:}\]
\[\text{Flaw found in test 1 and 2}\]
Evaluation
Case Study: Shopping Cart Web Applications

Customers

Order

Online Store

Cashier-as-a-Service

Pay

Trace Collection
Experiments and Results

- Target: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores

- Testbed: created 12 Paypal sandbox configurations

In total 3,145 test cases
Experiments and Results

- Target: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores

- Testbed: created 12 Paypal sandbox configurations

  1,253 “misuse” detected

In total 3,145 test cases

  1,892 were executed
Experiments and Results

- **Target**: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores

- **Testbed**: created 12 Paypal sandbox configurations

1,892 were executed

1,253 “misuse” detected

In total **3,145 test cases**

909 violation

983 not violations
Experiments and Results

- Target: 7 popular eCommerce Web Applications
  - Deployed by >13M online stores

- Testbed: created 12 Paypal sandbox configurations

In total **3,145** test cases
- **1,253** “misuse” detected
- **1,892** were executed
- **909** violation
  - **849** caused by bugs in the GUI
  - **90** caused by logic flaws
- **983** not violations
10 previously-unknown vulnerabilities

- Allowing to shop for free or pay less

<table>
<thead>
<tr>
<th>Application</th>
<th>Shop for free</th>
<th>Pay less</th>
<th>Session Fixation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbanteCart</td>
<td>✗</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Magento</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenCart</td>
<td></td>
<td>✗ ✗</td>
<td></td>
</tr>
<tr>
<td>osCommerce</td>
<td>✗</td>
<td>✗</td>
<td>CVE-2012-2991</td>
</tr>
<tr>
<td>PrestaShop</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TomatoCart</td>
<td>✗</td>
<td>✗ ✗</td>
<td>✗</td>
</tr>
<tr>
<td>CS-Cart</td>
<td>✗</td>
<td></td>
<td>CVE-2012-4934</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CVE-2013-0118</td>
</tr>
</tbody>
</table>
Conclusion
Conclusion

- Proposed a black-box technique to detect logic flaws in web applications
- Combined passive model inference and attacker pattern-based test case generation

- Developed a prototype
  - assessed against 7 popular eCommerce web applications

- Discovered 10 previously-unknown logic flaws
  - allow an attacker to shop for free or pay less
References


Thank you

Contact Information:

Giancarlo Pellegrino
gpellegrino@deeds.informatik.tu-darmstadt.de
Backup slides
## Results

<table>
<thead>
<tr>
<th>Applications</th>
<th>CaaS</th>
<th># Test Cases</th>
<th># TC Exec.</th>
<th>Property Violations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>due to Bugs</td>
</tr>
<tr>
<td>AbanteCart</td>
<td>Std</td>
<td>233</td>
<td>74</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>Exp</td>
<td>343</td>
<td>240</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>386</td>
<td>210</td>
<td>126</td>
</tr>
<tr>
<td>Magento</td>
<td>Exp</td>
<td>173</td>
<td>140</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>135</td>
<td>71</td>
<td>30</td>
</tr>
<tr>
<td>OpenCart</td>
<td>Exp</td>
<td>165</td>
<td>117</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>225</td>
<td>128</td>
<td>34</td>
</tr>
<tr>
<td>osCommerce</td>
<td>Exp</td>
<td>137</td>
<td>85</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>225</td>
<td>115</td>
<td>24</td>
</tr>
<tr>
<td>PrestaShop</td>
<td>Exp</td>
<td>137</td>
<td>85</td>
<td>-</td>
</tr>
<tr>
<td>TomatoCart</td>
<td>Exp</td>
<td>302</td>
<td>238</td>
<td>65</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>224</td>
<td>115</td>
<td>24</td>
</tr>
<tr>
<td>CS-Cart</td>
<td>Exp</td>
<td>600</td>
<td>347</td>
<td>313</td>
</tr>
<tr>
<td></td>
<td>Std</td>
<td>222</td>
<td>127</td>
<td>108</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>3145</strong></td>
<td><strong>1892</strong></td>
<td><strong>849</strong></td>
</tr>
</tbody>
</table>
osCommerce and AbanteCart: Shopping for Free
Understand the web application
  • Intended workflow and data flow

Design tests to violate workflow and data flow
  • E.g., reorder steps, replay tokens, …

Run tests and observe the results
### Problem

<table>
<thead>
<tr>
<th>Source code</th>
<th>Explicit Documentation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Black-box</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White-box</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Black-box</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Design verification</td>
</tr>
<tr>
<td></td>
<td>No</td>
<td>Black-box</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Design verification</td>
</tr>
</tbody>
</table>

- **White-box testing** [BalzarottiCCS07, FelmetsgerUSENIX10, ...]
  - Source code of WA may not be available → White-box not applicable!!

- **Design verification** [LoweCSF97, ArmandoCSF07, ...]
  - Specification of WA may not be available → DV not applicable!

- **Black-box testing, e.g., web scanners** [DoupèDIMVA10, WangS&P11, WangS&P12]
  - Cannot automatically detect logic flaws

> **Testing for logic flaws is done manually**
Workflow Patterns

Traces:
\[ \pi_1 = \langle a, b, a, c, d, e, f, e \rangle \]
\[ \pi_2 = \langle a, c, d, e, f, e \rangle \]
Workflow Patterns

Traces:

\[ \pi_1 = \langle a, b, a, c, d, e, f, e \rangle \]
\[ \pi_2 = \langle a, c, d, e, f, e \rangle \]

Model:

- TrWP: Trace Waypoints
- St: Singleton Nodes
- Rp: Repeatable Operations
- MWP: Model Waypoints

Diagram showing the sequence of operations with labels for each node and operation type.