Frameworks & Security
How web frameworks kill your static security scans

Christian Hang
Armorize Technologies
chris@armorize.com

OWASP
AppSec Research 2010

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document
under the terms of the OWASP License.

The OWASP Foundation
http://www.owasp.org
Motivation

- Web Frameworks are omnipresent
- Frameworks extend application model
- Static code analysis hits technology limits
- Can frameworks be addressed with SCA?
- Can it be done in open & extensible way?
Static Code Analysis

- Compile time scan on code or binaries
- Mostly data-flow oriented
- Often provides traces and points to LOC
- Potentially integrated in dev. processes
SCA Technology Limits

- SCA is compile-time ⇒ no runtime data
- Runtime types unknown ⇒ flow unclear
- Execution environment not accessible
- Code might be incomplete
- Application model must be known
How about Frameworks?

- Web Frameworks want to help you
  - Figure out action based on URL
  - Prepare user input to be easily accessible
  - Separate Business Logic and Views
- “Magic” happening in the background
- Runtime behavior that’s opaque
Example: A Struts Request

1. registration.jsp
2. struts-config.xml
3. RegistrationAction.java
4. registration-Success.jsp
5. forward name="success" path="/registrationSuccess.jsp"
6. action attribute="registrationForm" input="/form/registration.jsp" scope="request" type="...action.RegistrationAction"
What’s the problem?

- Reflection
  - program can modify its structure and behavior at runtime
  - Static = Runtime type unknown

Where do I go from here?

- Flow trace is
  - Potential for False broken/ unclear Negative/ Positive

```java
((Action) Class.forName(?????????? ).newInstance()).execute(req);
```
What’s the problem?

- Invocation Sequence
- Cross-Context-Propagation

```java
Servlet
public void doGet(HttpServletRequest req, HttpServletResponse resp) {
    String user = req.getParameter("user");
    req.setAttribute(??????????????????, user);

    getServletConfig().getServletContext().getRequestDispatcher(???????????????).forward(req,resp);
}

JSP
<p><%= request.getAttribute(???????????????) %></p>
```
SCA Scan results

- Tainted Source
  ```java
  req.getParameter("user");
  ```
- Dataflow path
  ```java
  String user = req.getParameter("user");
  req.setAttribute(???????????????????, user);
  ```
- Is this a sink?
  ```java
  <%= request.getAttribute(??????????????) %>
  ```
- Assume attribute is clean / tainted
- Potential for False Negative / Positive
Summary: Flow Disruptions

- URL invoke Actions
  - Not obvious from source code: See XML
- Actions forward to Views
  - Not obvious from source code: See XML
- Views output data from Action
  - Cross-Context Propagation
Challenges

- Struts: XML key to understanding app.
- SCA tool must model framework
- Which frameworks to support?
- How about your home-grown one?
Possible Solutions?

- Require user to hardcode configuration 😞
- Tools hardcode support for framework 😞
- Dynamically translate magic into code 😊
Glue Code Generation

- Resolve reflection ambiguity

```xml
<struts-config>
  <form-beans>
    <form-bean name="registrationForm" type="com.domain.form.RegistrationForm" />
  </form-beans>
  <action-mappings>
    <action attribute="registrationForm" input="registrationInput.jsp"
             name="registrationForm" path="/registration" scope="request"
             type="com.domain.action.RegistrationAction">
      <forward name="success" path="/registrationSuccess.jsp" />
      <forward name="fail" path="/registrationFail.jsp" />
    </action>
  </action-mappings>
</struts-config>

RegistrationAction ra = new RegistrationAction();
ActionForward fwd = ra.execute(...);
```
Glue Code Generation

Connect controller & views

```xml
<struts-config>
  <form-beans>
    <form-bean name="registrationForm" type="com.domain.form.RegistrationForm" />
  </form-beans>
  <action-mappings>
    <action attribute="registrationForm" input="registrationInput.jsp"
        name="registrationForm" path="/registration" scope="request"
        type="com.domain.action.RegistrationAction">
      <forward name="success" path="/registrationSuccess.jsp" />
      <forward name="fail" path="/registrationFail.jsp" />
    </action>
  </action-mappings>
</struts-config>

RegistrationAction ra = new RegistrationAction();
ActionForward fwd = ra.execute(...);
if (...) {
  req.getRequestDispatcher("registrationSuccess.jsp").forward(req, res);
} else {
  req.getRequestDispatcher("registrationFail.jsp").forward(req, res);
}
```
Simple & Effective Workaround

- No impact on implementation or code
- Several Options
  - Standalone (3rd party) infrastructure
  - Bundled with tool
- Not perfect, but easily extendable
- Applicable to “home-grown” frameworks
- Extends coverage of automatic analysis
RegistrationAction ra = new RegistrationAction();
ActionForward fwd = ra.execute(...);

public ActionForward execute(ActionMapping map, ...) {
    String firstname = req.getParameter("firstname");
    req.setAttribute("new_user", firstname);
    return map.findForward("success");
}

if (...) {
    req.getRequestDispatcher("registrationSuccess.jsp").forward(req, res);
} else {
    req.getRequestDispatcher("registrationFail.jsp").forward(req, res);
}

Welcome <%= request.getAttribute("new_user") %>!
Conclusion

- Web framework make static scanning hard
- SCA tools can scan frameworks effectively
- On the fly “translation” increases coverage
- Possibility to handle this in cross-tool way