"… use the source …"

How secure your web framework is?

Based on Apache Struts 2

@lukaszlenart
@TheApacheStruts
lukaszlenart@apache.org

Agenda

- About me
- What is the Apache Struts 2
- Hacking the framework
- What about the others
- Home work
- Q&A

About me

- Apache Struts 2 Lead & Member of ASF
- Creative Software Engineer @ 

- Blogger, @lukaszlenart
- IntelliJ IDEA addict ☺
- Husband, father, biker ☺

:-)

Struts 1 …. is dead, baby ☺

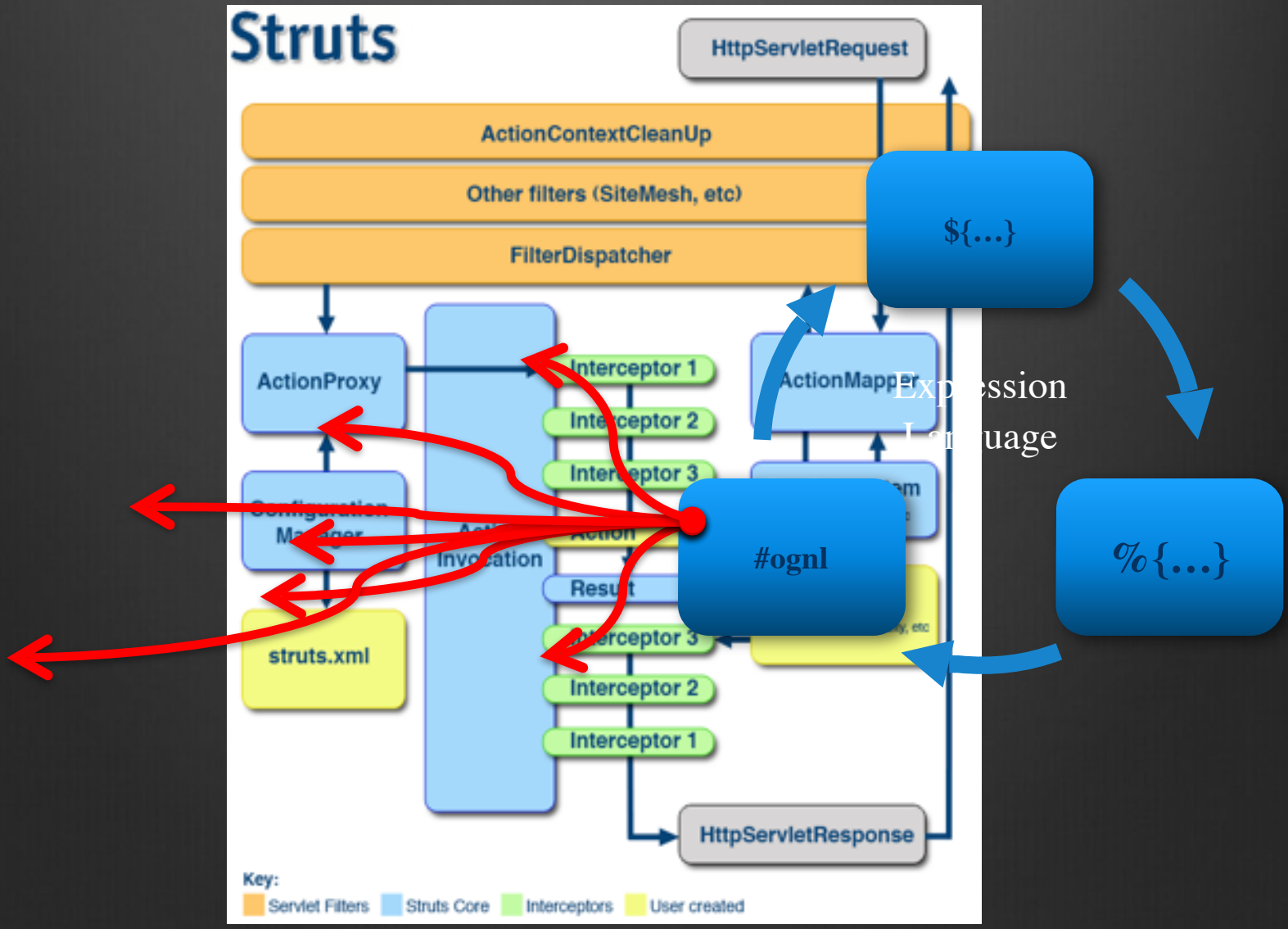
- Struts 1 reached EOL! (over a year ago!)
- Struts 2 is a new kid on the block
  - No single line shared with Struts 1
  - No form beans, no session-scoped actions
  - Pure POJOs, Interface steering
  - Strongly interceptor oriented
  - Highly extendable – lots of plugins
  - Designed to be customisable
  - Powerful OGNL expression language

The King is dead
Long live the King!

Struts 2 is now the Apache Struts

With great power…

How does it work?



Expressions are everywhere

struts.xml

```
<action name="index" class="org.demo.MyAction" method="index">
    <result name="input">index.jsp</result>
    <result type="redirect">${actionName}</result>
</action>
```

index.jsp

```
<s:form action="submitAddressesInfo" namespace="/conversion">
    <s:iterator value="%{new int[3]}" status="stat">
        <s:textfield label="%{'Address '+#stat.index}"
                     name="%{'addresses(\\'id'+#stat.index+'\\').address'}" />
    </s:iterator>
    <s:submit cssClass="btn btn-primary"/>
</s:form>
```

IndexAction.properties

```
HelloWorld.message= Struts is up and running ...
requiredstring = ${getText(fieldName)} is required.
password = Password
username = User Name
Missing.message =  This feature is under construction.
```

Hacking the framework

. . . .

be the bad guy

S2-006 aka Client side code injection

- When Dynamic Method Invocation is enabled action name is generated based on the provided request
- Non-existing action will generate an error page with injected client code
  - Issue is specific to Weblogic server
- http://struts.apache.org/2.x/docs/s2-006.html

S2-006 aka Client side code injection - example

/HelloWorld.action?action%3Alogin!login%3AcantLogin%3Cscript%3Ealert%28window.location%29%3C%2Fscript%3E%3Dsome_value=Submit

S2-006 aka Client side code injection - solution

- Disable DMI
  - <constant name="struts.enable.DynamicMethodInvocation" value="false" />
- Upgrade to Struts 2.2.3
- Don’t use Weblogic ;-)

S2-008 aka Remote Command Execution

- Conversion error is evaluated as an expression
- Cookie name is evaluated as an expression
- With "!" (bang) you can access any public method of action
  - Only when Dynamic Method Invocation is set to true, is set to true by default
- http://struts.apache.org/2.x/docs/s2-008.html

S2-008 aka Remote Command Execution – example

- /hello.action?id='%2b(new Object())%2b’
- Cookie: @java.lang.Runtime@getRuntime().exec()=1
- /mywebapp/recover!getPassword.action

S2-008 aka Remote Command Execution - solution

- Disable DMI
  - <constant name="struts.enable.DynamicMethodInvocation" value="false" />
- Review action's public methods
- Use Strict DMI – list of allowed methods
- DMI disabled by default as from Struts 2.3.1
- Upgrade to Struts 2.3.1!

S2-009 aka RCE strikes back

- An arbitrary code can be executed on server
  - Encoded value of parameter is parsed as an OGNL expression

- http://struts.apache.org/2.x/docs/s2-009.html

S2-009 aka RCE strikes back - example

/action?foo=
%28%23context[%22xwork.MethodAccessor.denyMethod
Execution%22]%3D+new+java.lang.Boolean%28false
%29,%20%23_memberAccess[%22allowStaticMethodAcc
ess%22]%3d+new+java.lang.Boolean%28true
%29,%20@java.lang.Runtime@getRuntime%28%29.exec
%28%27mkdir%20/tmp/PWNAGE%27%29%29%28meh
%29&z[%28foo%29%28%27meh%27%29]=true

S2-009 aka RCE strikes back - solution

- Stronger pattern for parameter names
  - OGNL only sets value, does not evaluate it
- Workaround
  - add a filter to filter out all the suspicious looking parameters/headers
- Upgrade to Struts 2.3.1.2

S2-011 aka DoS

- Denial of Service
  - Long request parameter name is evaluated by OGNL and consumes significant CPU cycle

- http://struts.apache.org/2.x/docs/s2-011.html

S2-011 aka DoS - example

- POST /home veryveryveryevenveryveryveryveryveryveryveryveryveryevenevenveryveryveryveryloooooooooooooongpramaterename=1
  - 300 request
  - parameter name length = 1000000

S2-011 aka DoS - solution

- Add parameter name length limit
  - By default 100 characters
  - User can change the limit
- Workaround
  - add a filter to filter out all the parameters longer than xxx
- Upgrade to Struts 2.3.4.1

S2-016 aka RCE never dies!

- An arbitrary code can be executed on server
  - action: / redirect: / redirectAction: allow remote command execution

- http://struts.apache.org/2.x/docs/s2-016.html

S2-016 aka RCE never dies! example

/save.action?redirect:%25{(new+java.lang.ProcessBuilder(new+java.lang.String[]{‘command','goes','here'})).start()}

S2-016 aka RCE never dies! solution

- Support for action: prefix disabled by default and removed expression evaluation
  - removed support for redirect: and redirectAction:
- Workaround
  - add a filter to filter out all the parameters prefixed with action:, redirect: or redirectAction:
- Upgrade to Struts 2.3.15.1

Sx-xxx aka more to come

. . . .

You never know what future will bring for us ☺

have one's finger on the pulse

Prior Releases

As a courtesy, we retain archival copies of the website for releases that initially were considered "General Availability" but which has been reclassified as "Not recommended" since they contain security issues

Release	Release Date	Vulnerability	Version Notes
Struts 2.3.16	8 December 2013	S2-020	Version notes
Struts 2.3.15.3	15 October 2013	S2-020	Version notes
Struts 2.3.15.2	16 July 2013	S2-018, S2-020	Version notes
Struts 2.3.15.1	16 July 2013	S2-019, S2-020	Version notes
Struts 2.3.15	22 June 2013	S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.14.3	3 June 2013	S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.14.2	22 May 2013	S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.14.1	22 May 2013	S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.14	11 April 2013	S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.12	6 March 2013	S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.8	22 December 2012	S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.7	19 November 2012	S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.4.1	13 August 2012	S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.4	12 May 2012	S2-010, S2-011, S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.3	16 April 2012	S2-010, S2-011, S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes
Struts 2.3.1.2	22 January 2012	S2-010, S2-011, S2-012, S2-013, S2-014, S2-015, S2-016, S2-017, S2-018, S2-019, S2-020	Version notes

What about the others

Apache » Tomcat : Vulnerability Statistics

Vulnerabilities (116) CVSS Scores Report Browse all versions Possible matches for this product Related Metasploit Modules

Related OVAL Definitions : Vulnerabilities (126) Patches (42) Inventory Definitions (1) Compliance Definitions (0)

Vulnerability Feeds & Widgets

Vulnerability Trends Over Time

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2000	3														
2001	4						1								
2002	12	4		1			1	1		1	3				
2003	7	2	1				2			1					
2005	7	2					2			1	3				
2006	1														
2007	17						9	2			3		1		1
2008	9						2	2		1	3				1
2009	8	1					1	1		1	4	1			
2010	8	1		1			2	2		1	2				
2011	14	2					1	1		7	2	1			
2012	15	5								9	1		1		
2013	4	1									1		1		
2014	7	2								1	2				
Total	116	20	1	2			21	9		23	24	2	3		2
% Of All		17.2	0.9	1.7	0.0	0.0	18.1	7.8	0.0	19.8	20.7	1.7	2.6	0.0	

Don’t be fool!

Use SecurityManager!

Home work

1. Check how vulnerable your current web framework is
2. Find a security vulnerability, try to inject JavaScript, etc.
3. Report back to the project team

Q&A

This is the end, questions?

https://github.com/lukaszlenart/how-secure-your-framework-is

@lukaszlenart
@TheApacheStruts
lukaszlenart@apache.org