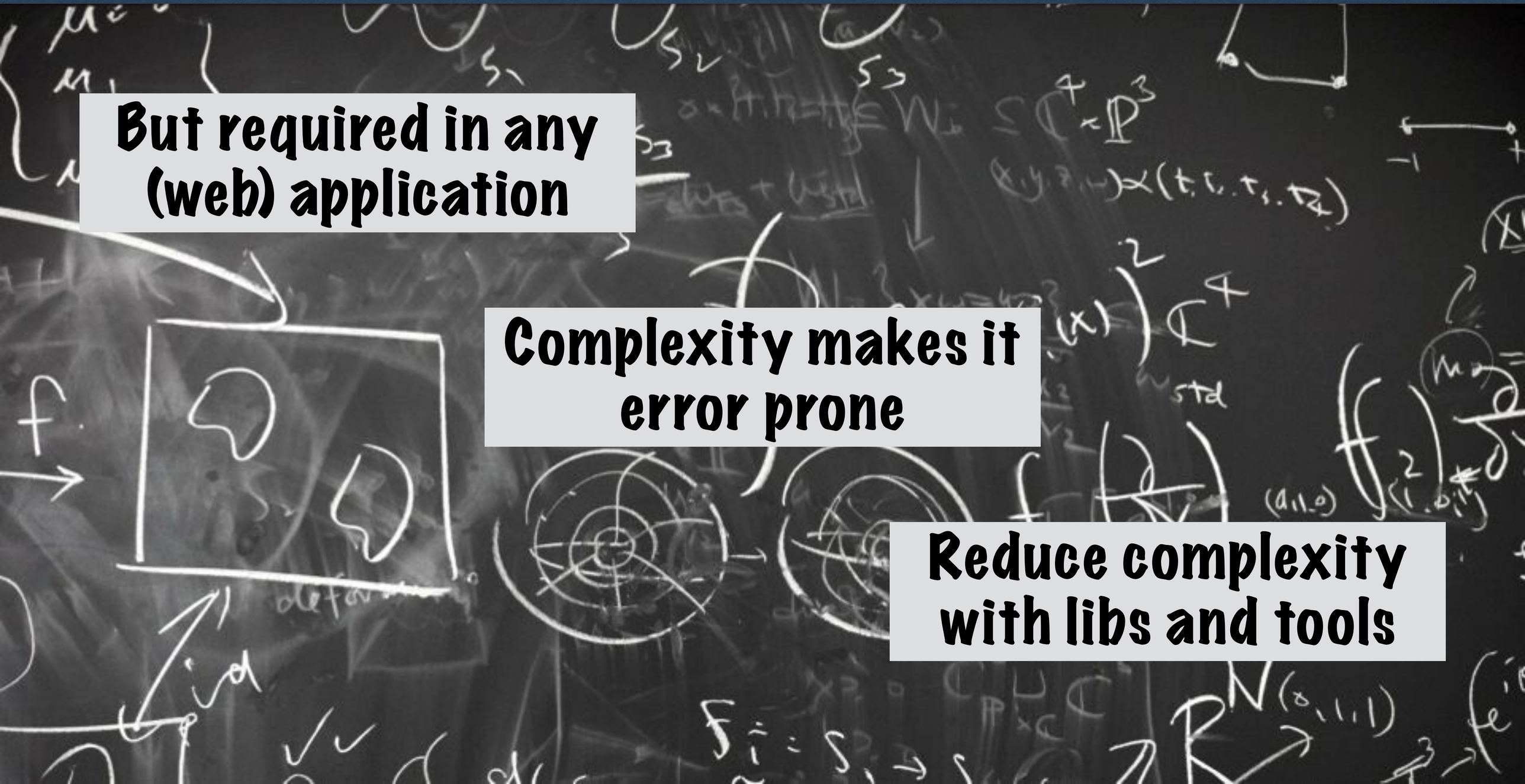
7 security tools and libraries every developer should know about

JavaOne 2014

Dominik Schadow | bridgingIT

Security is a complex topic



But required in any (web) application

Complexity makes it error prone

Reduce complexity with libs and tools

Spoilt for choice: frameworks and libraries

Security has to be tested like any other functionality

My seven tools and libs

Microsoft Threat Modeling Tool

OWASP Java Encoder

OWASP ZAP

Recx Security Analyser

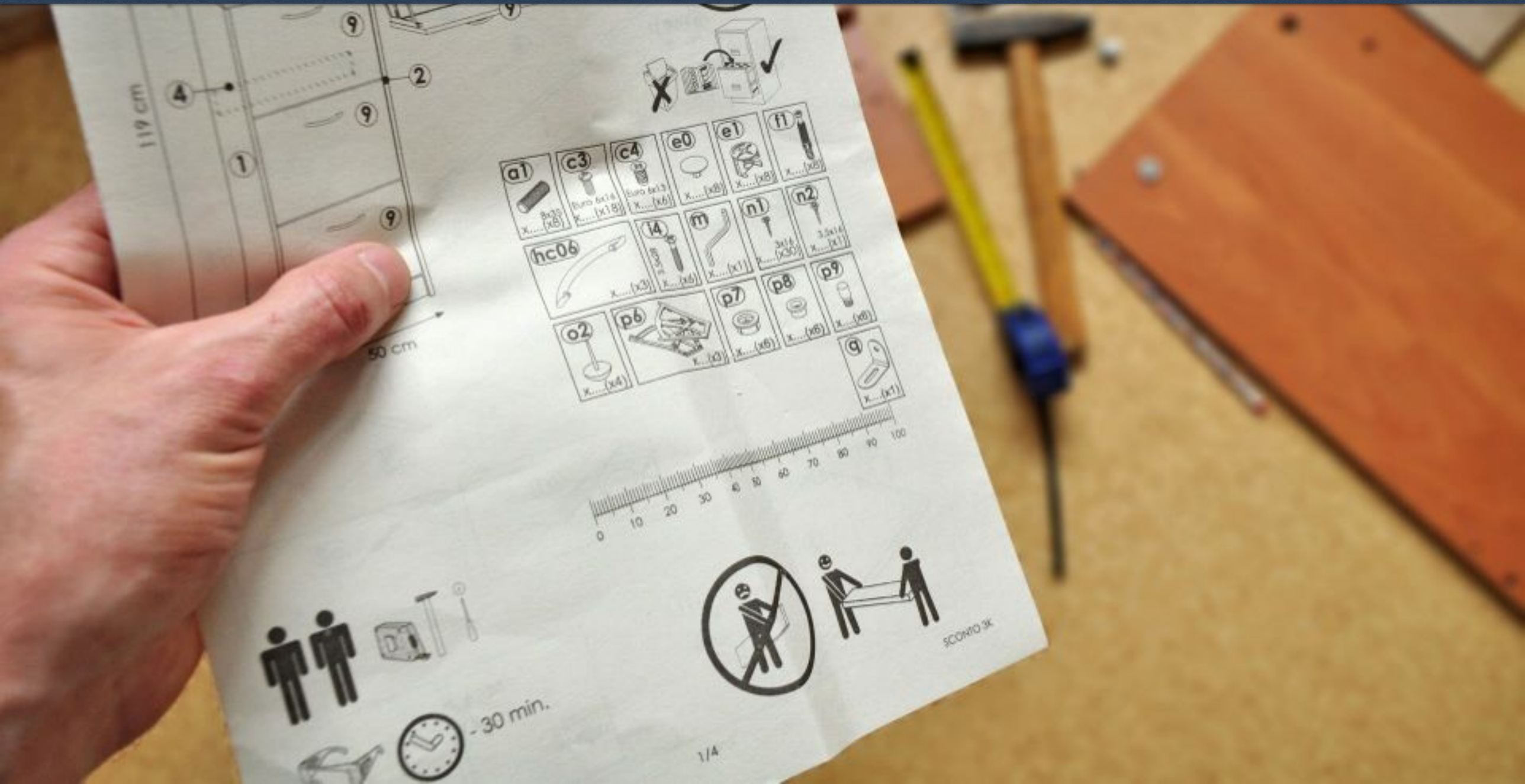
Jasypt

Keyczar

OWASP Dependency Check

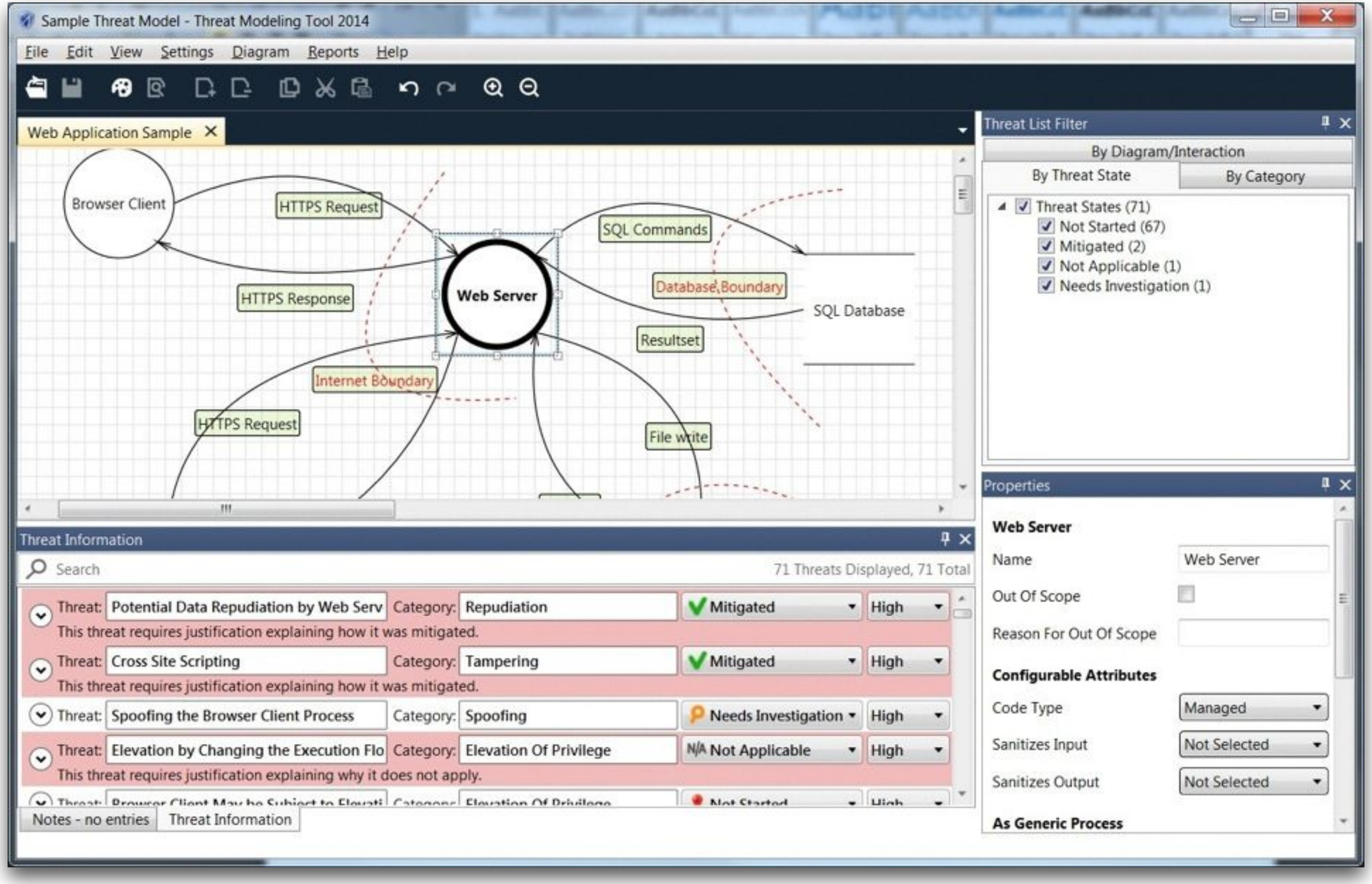
What about the well-known all-rounder frameworks?

So you are developing a new web application?!



Identify all incoming and outgoing data

Microsoft Threat Modeling Tool 2014



S **poofing**

Pretending to be somebody else

T **ampering**

Modifying data that should not be modifiable

R **epudiation**

Claiming someone didn’t do something

I **nformation Disclosure**

Exposing information

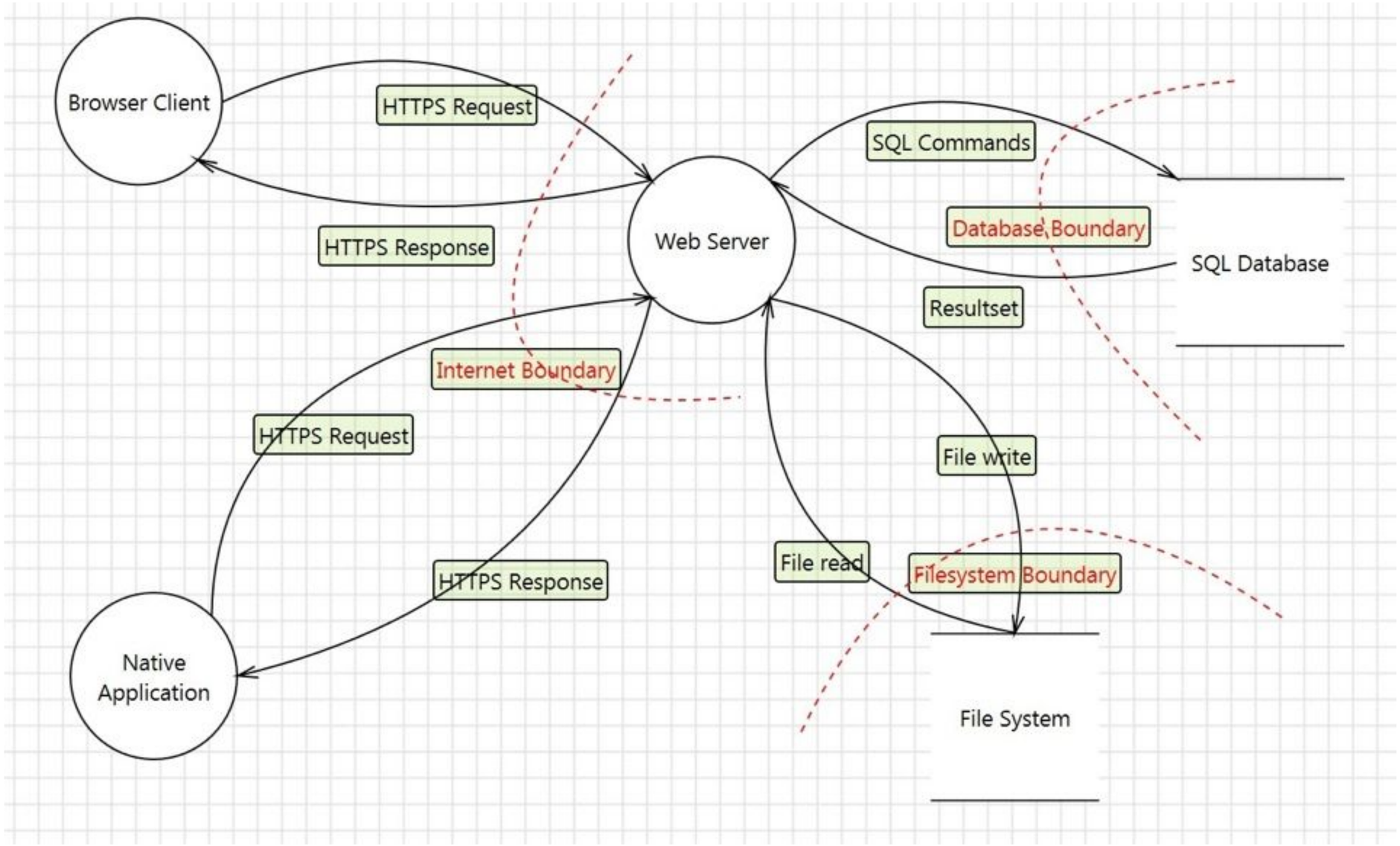
D **enial of Service**

Preventing a system from providing service

E **levation of Privilege**

Doing things that one isn’t supposed to do

Model your system and address the essential threats



Microsoft Threat Modeling Tool features

Participating systems and their boundaries

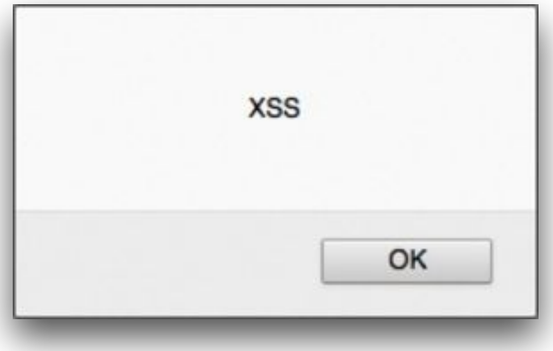
- Design and analysis view
- Security issue identification

- Issue tracker integration
- Report generation

Now you know all data entering and leaving your system

Escape all untrusted data for the proper context

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



For HTML

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

For CSS

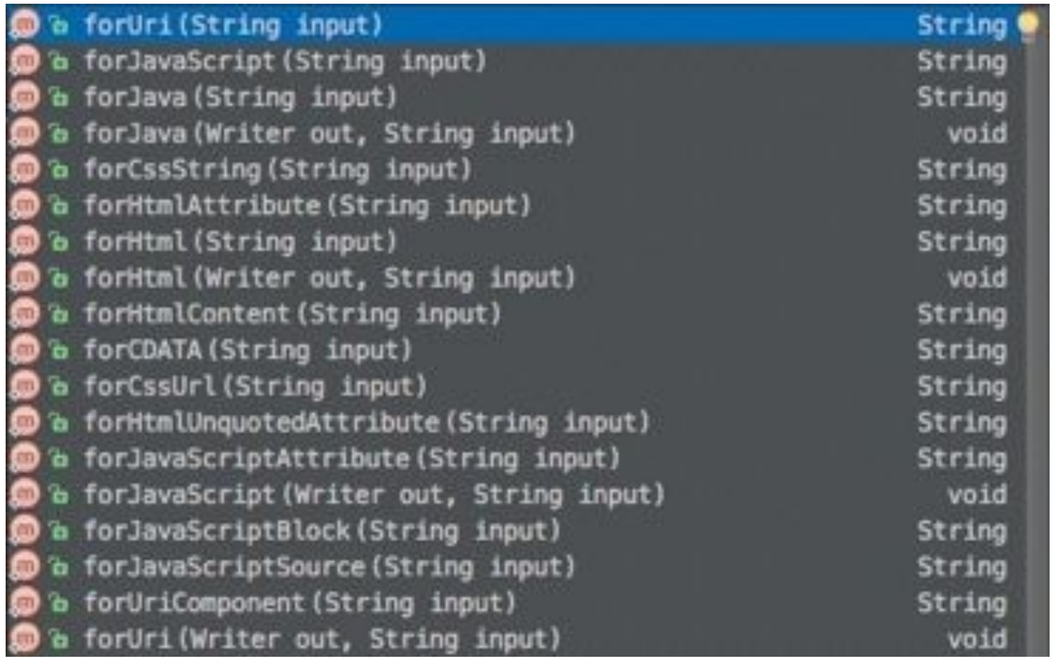
```
\3cscript\3e alert(\22XSS\22)\3c\2fscript\3e
```

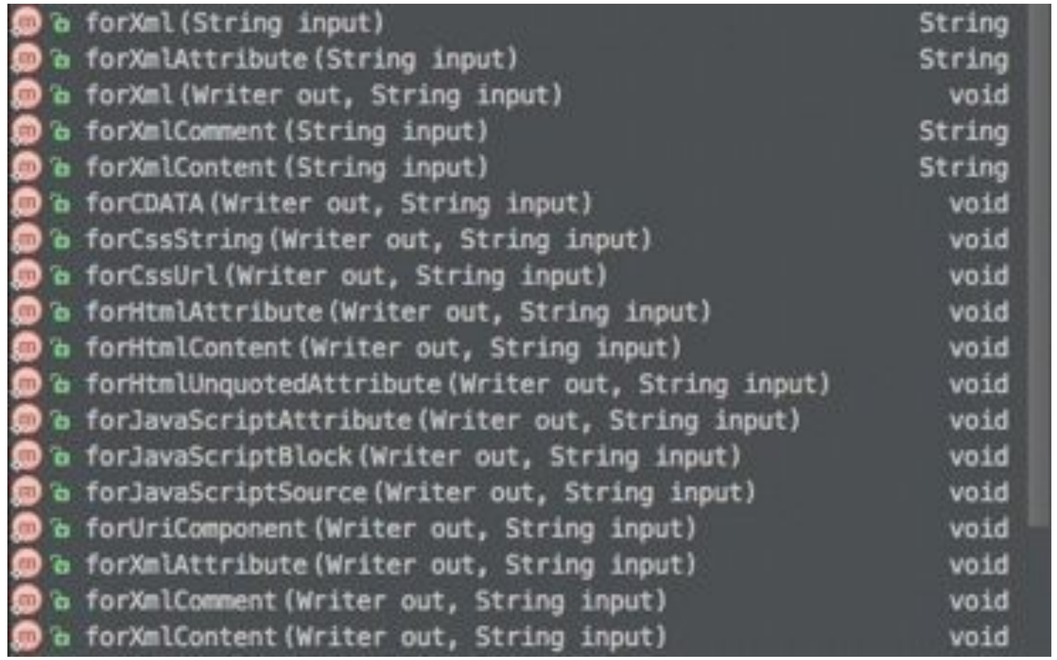
For XML

```
&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);
```





Escape directly inside JavaServer Pages

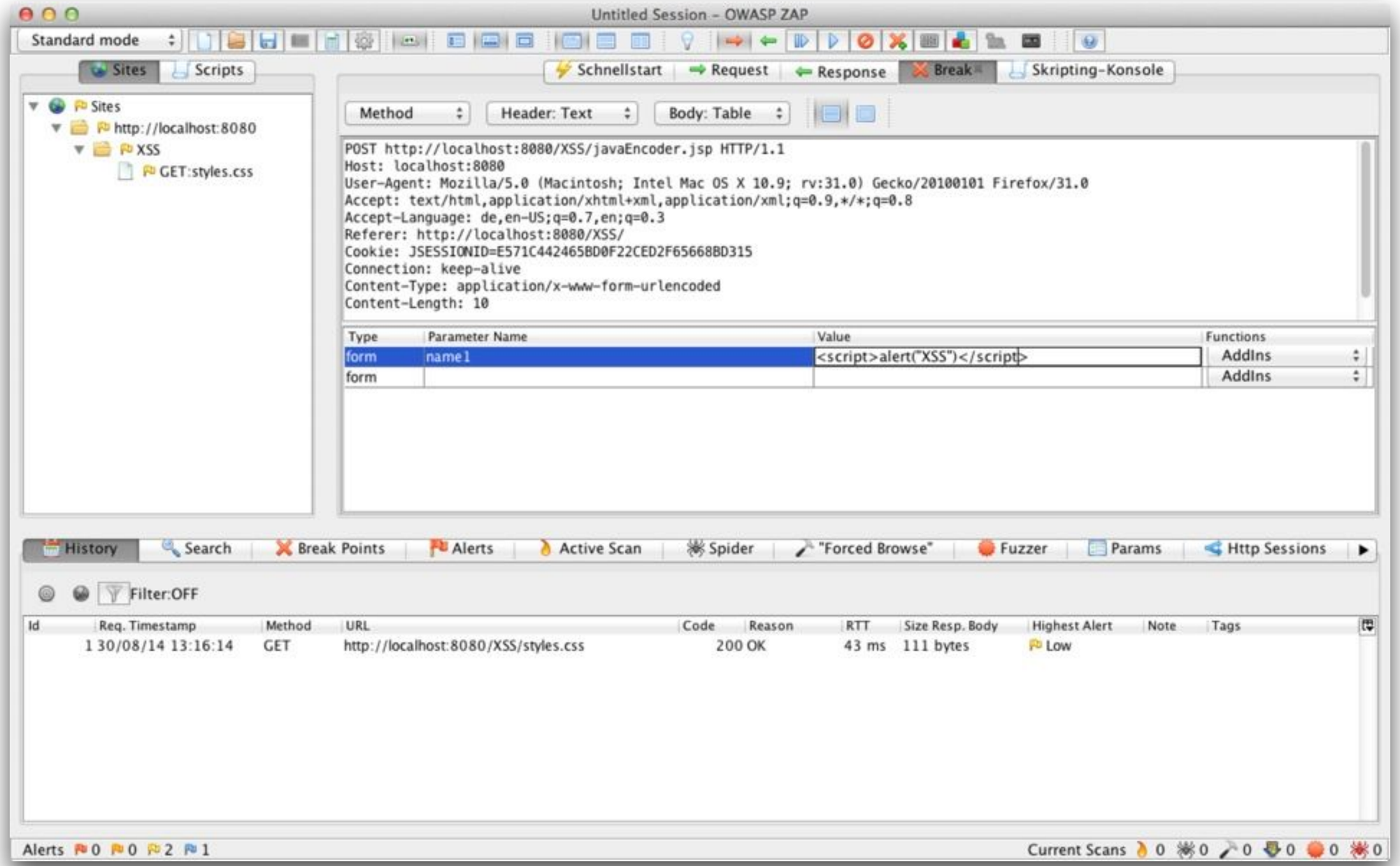
```
<%@ page language="java" contentType="text/html;
      charset=UTF-8" pageEncoding="UTF-8"%>
<%@ page import="org.owasp.encoder.Encode" %>
<%@ taglib prefix="e" uri="https://www.owasp.org/
      index.php/OWASP_Java_Encoder_Project" %>

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

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

Never ever do the escaping yourself! Never!

Test your validation and escaping boundaries

Intercept and manipulate HTTP requests and responses

OWASP Zed Attack Proxy (ZAP) and FoxyProxy



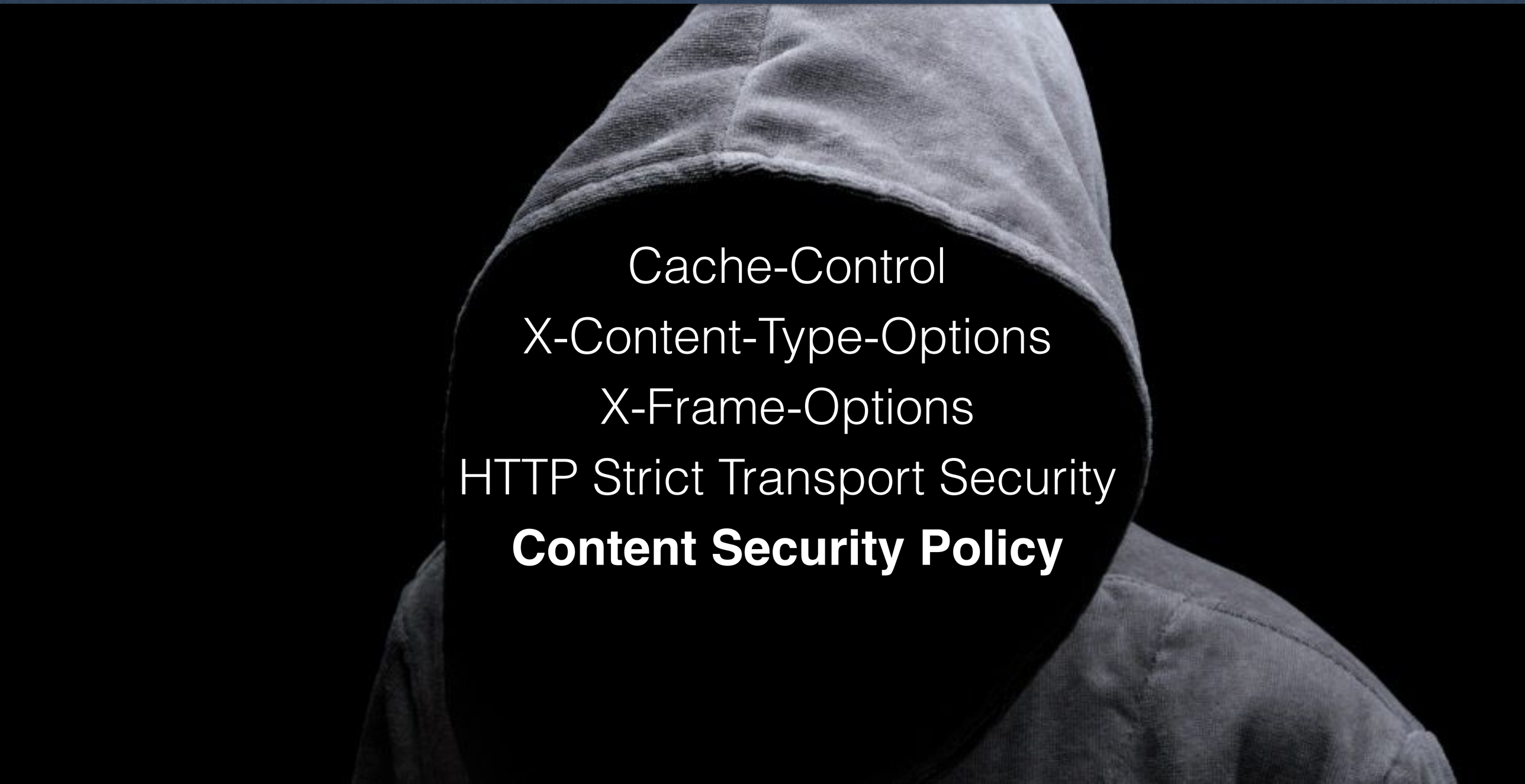
Intercepting proxy

- Active and passive scanners
- Forced Browsing
- Fuzzer
- Spider

- HTTP Session Management

Use the browser as part of your defense-in-depth strategy

Tell the browser how to secure your web application



Cache-Control

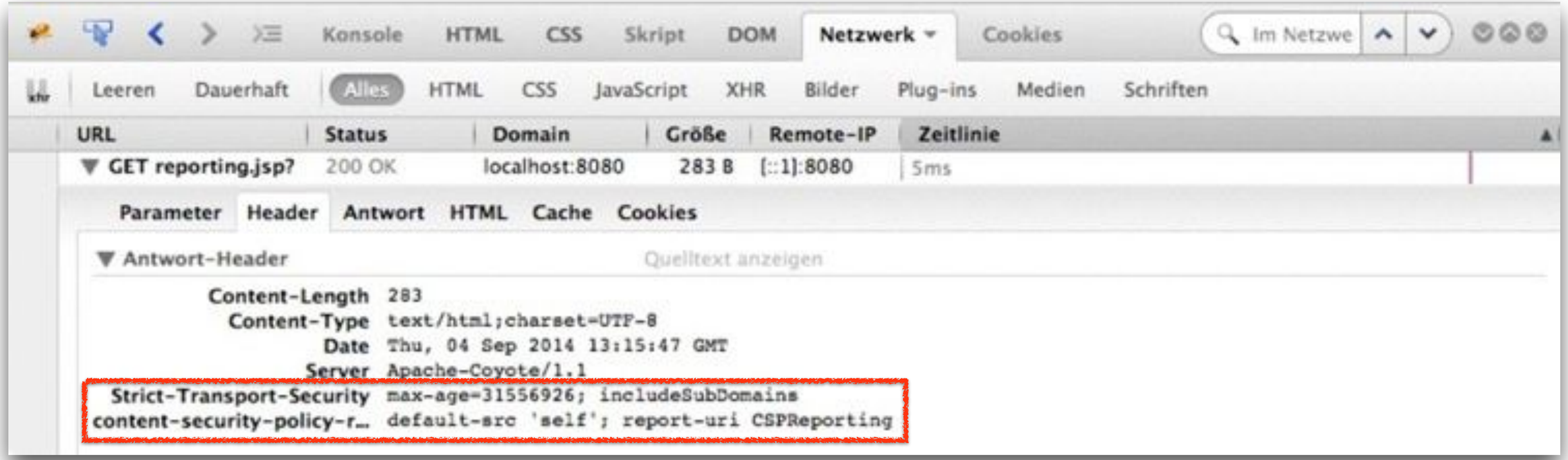
X-Content-Type-Options

X-Frame-Options

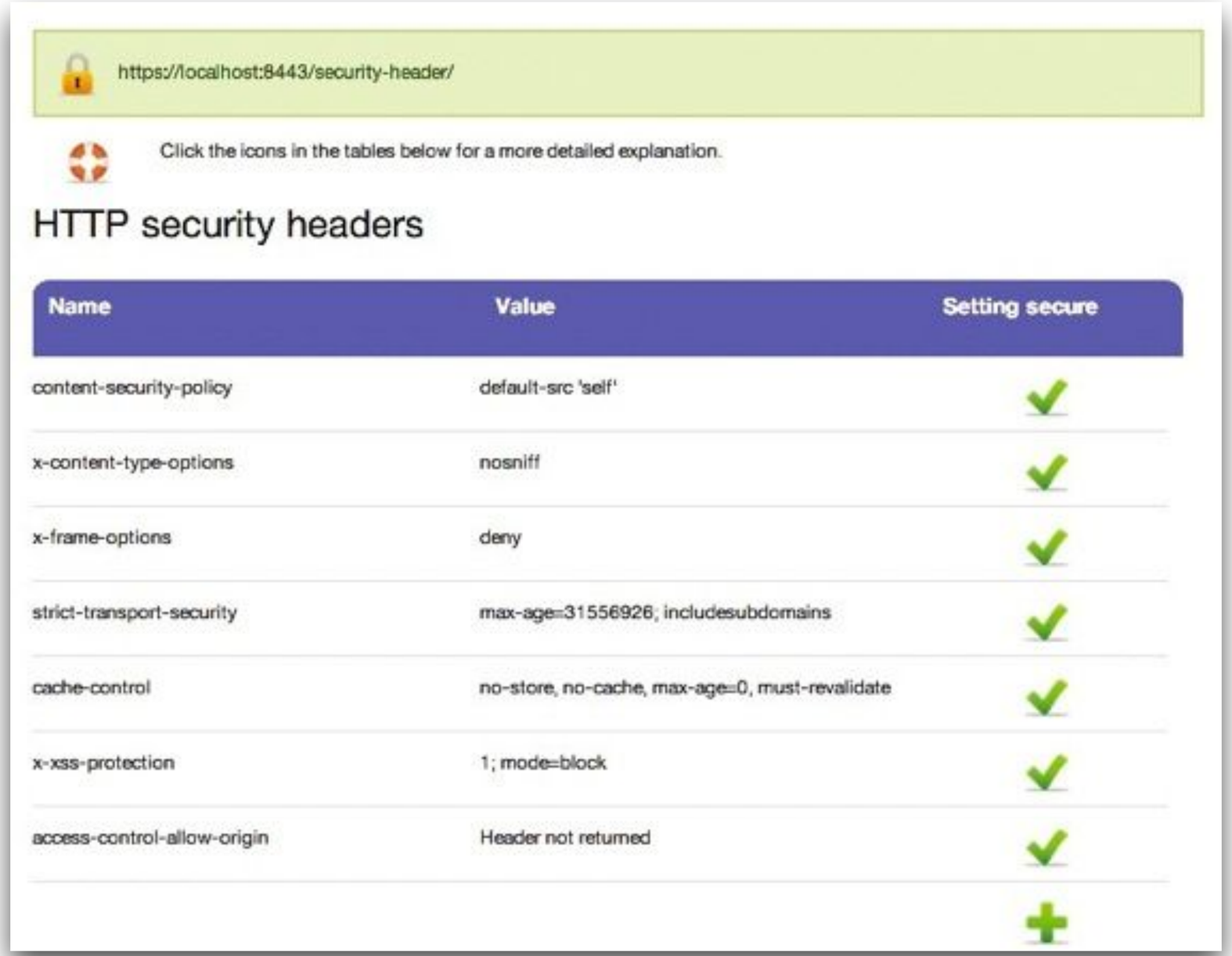
HTTP Strict Transport Security

Content Security Policy

Verify headers manually with Firebug



Recx Security Analyser

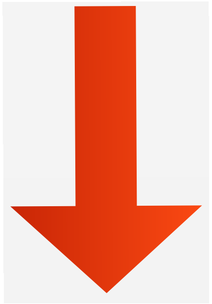


DEMO

Keep your configuration passwords confidential

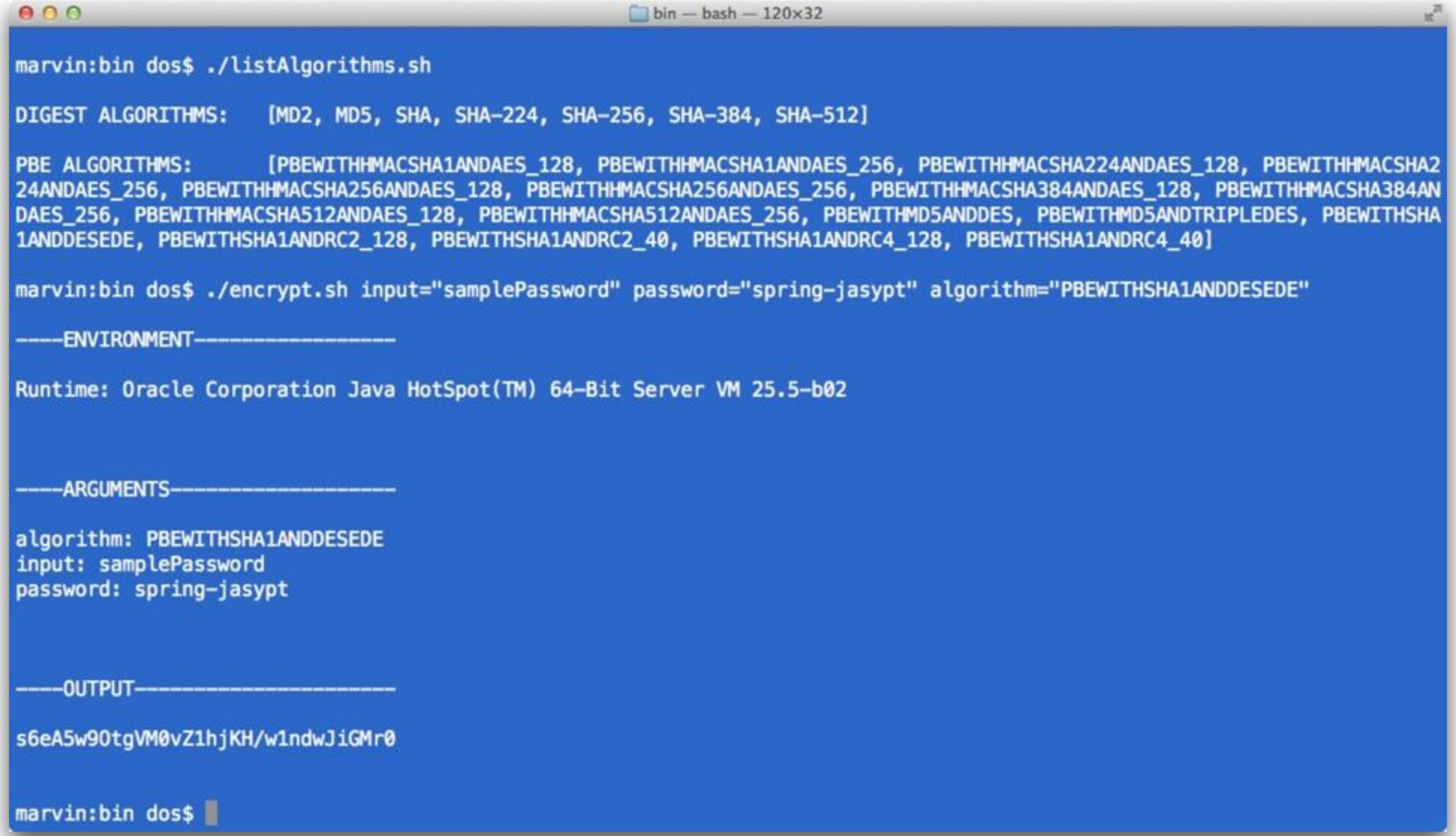
Typical DataSource configuration in a Spring XML file

```
<context:property-placeholder
  location="classpath:spring.properties" />

<bean id="dataSource" class="org...BasicDataSource">
  <!-- … -->
  <property name="url" value="${jdbc.url}"/>
  <property name="username" value="${jdbc.un}"/>
  <property name="password" value="${jdbc.pw}"/>
</bean>
```



```
jdbc.driver=org.hsqldb.jdbcDriver
jdbc.url=jdbc:hsqldb:mem:sampleDB
jdbc.un=sampleUser
jdbc.pw=samplePassword
```

Jasypt


bin — bash — 120×32
marvin:bin dos$ ./listAlgorithms.sh
DIGEST ALGORITHMS: [MD2, MD5, SHA, SHA-224, SHA-256, SHA-384, SHA-512]
PBE ALGORITHMS: [PBEWITHHMACSHA1ANDAES_128, PBEWITHHMACSHA1ANDAES_256, PBEWITHHMACSHA224ANDAES_128, PBEWITHHMACSHA224ANDAES_256, PBEWITHHMACSHA256ANDAES_128, PBEWITHHMACSHA256ANDAES_256, PBEWITHHMACSHA384ANDAES_128, PBEWITHHMACSHA384ANDAES_256, PBEWITHHMACSHA512ANDAES_128, PBEWITHHMACSHA512ANDAES_256, PBEWITHMD5ANDDES, PBEWITHMD5ANDTRIPLEDES, PBEWITHSHA1ANDDESEDE, PBEWITHSHA1ANDRC2_128, PBEWITHSHA1ANDRC2_40, PBEWITHSHA1ANDRC4_128, PBEWITHSHA1ANDRC4_40]
marvin:bin dos$ ./encrypt.sh input="samplePassword" password="spring-jasypt" algorithm="PBEWITHSHA1ANDDESEDE"
-----ENVIRONMENT-----
Runtime: Oracle Corporation Java HotSpot(TM) 64-Bit Server VM 25.5-b02
-----ARGUMENTS-----
algorithm: PBEWITHSHA1ANDDESEDE
input: samplePassword
password: spring-jasypt
-----OUTPUT-----
s6eA5w9OtgVM0vZ1hjKH/w1ndwJiGMr0
marvin:bin dos$

Replace plaintext password with encrypted password

```
jdbc.pw
=
ENC(s6eA5w9OtgVM0vZ1hjKH/w1ndwJiGMr0)
```

Provide the encryption password at server start

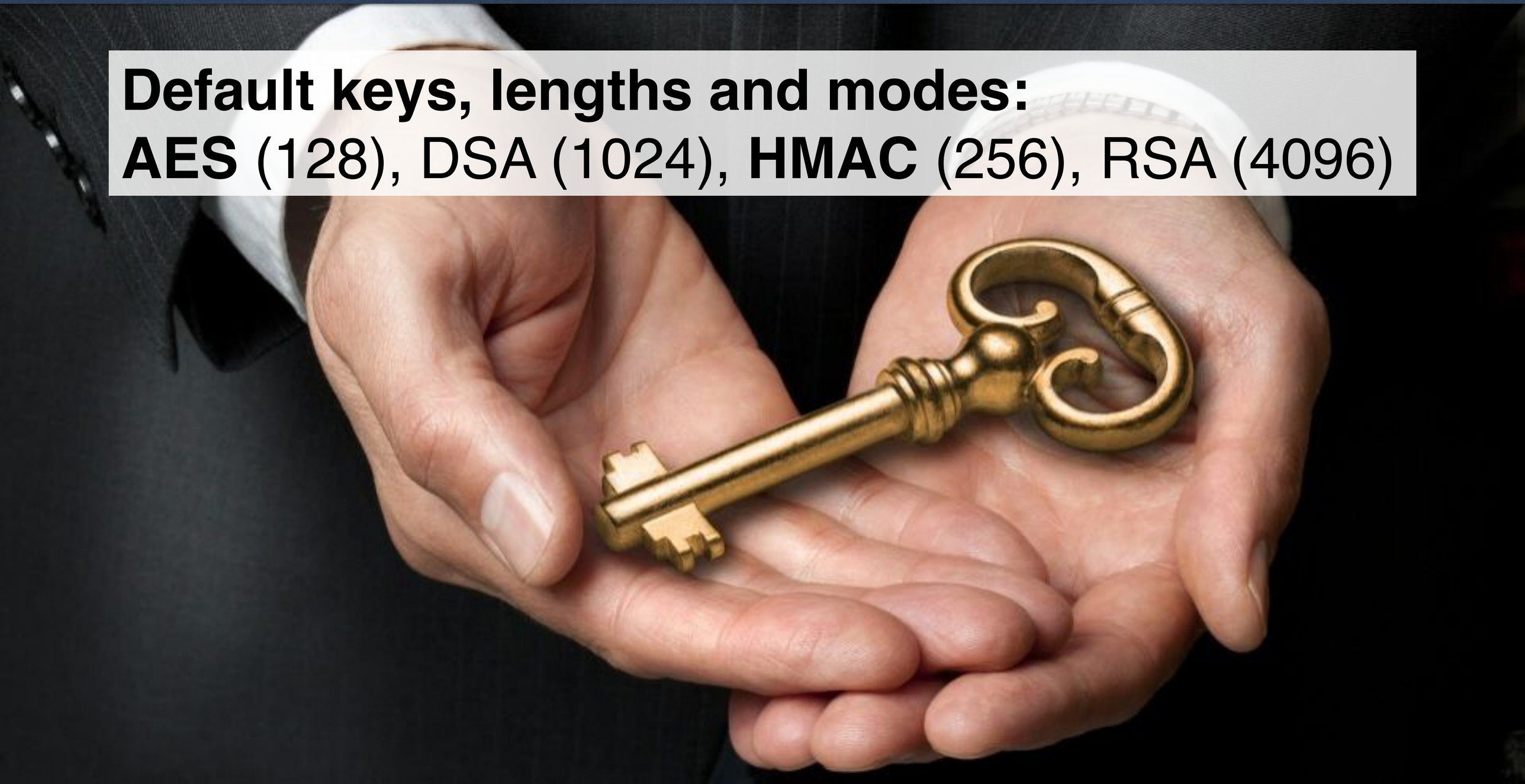
Encrypt all sensitive data with safe algorithms

AES Modes
ECB, CBC, CFB,
OFB, CTR, …

Which algorithm, mode, padding, …?

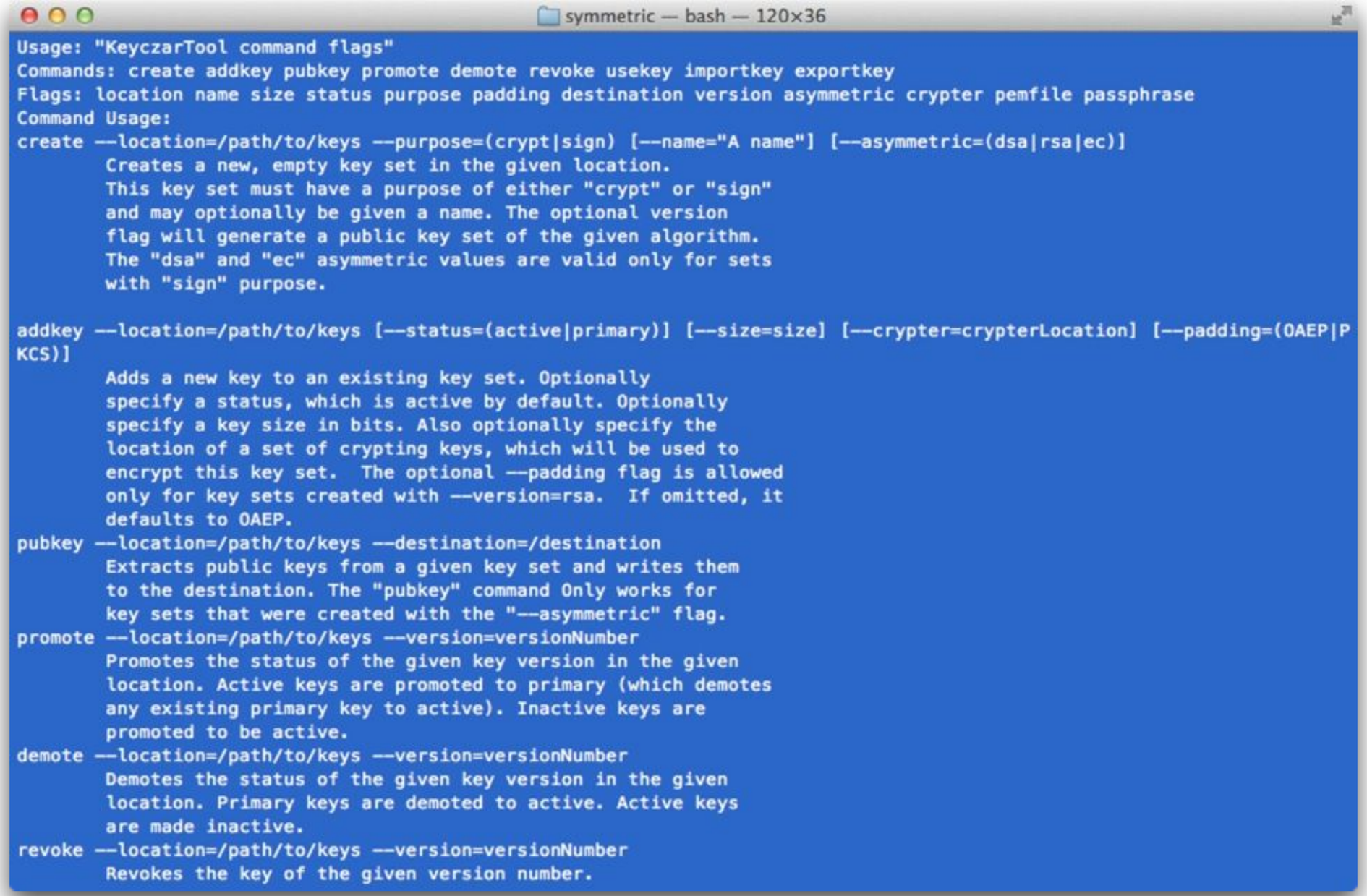
Swap all keys regularly like any other password

Keyczar



Default keys, lengths and modes:
AES (128), DSA (1024), **HMAC** (256), RSA (4096)

KeyczarTool for command line key management



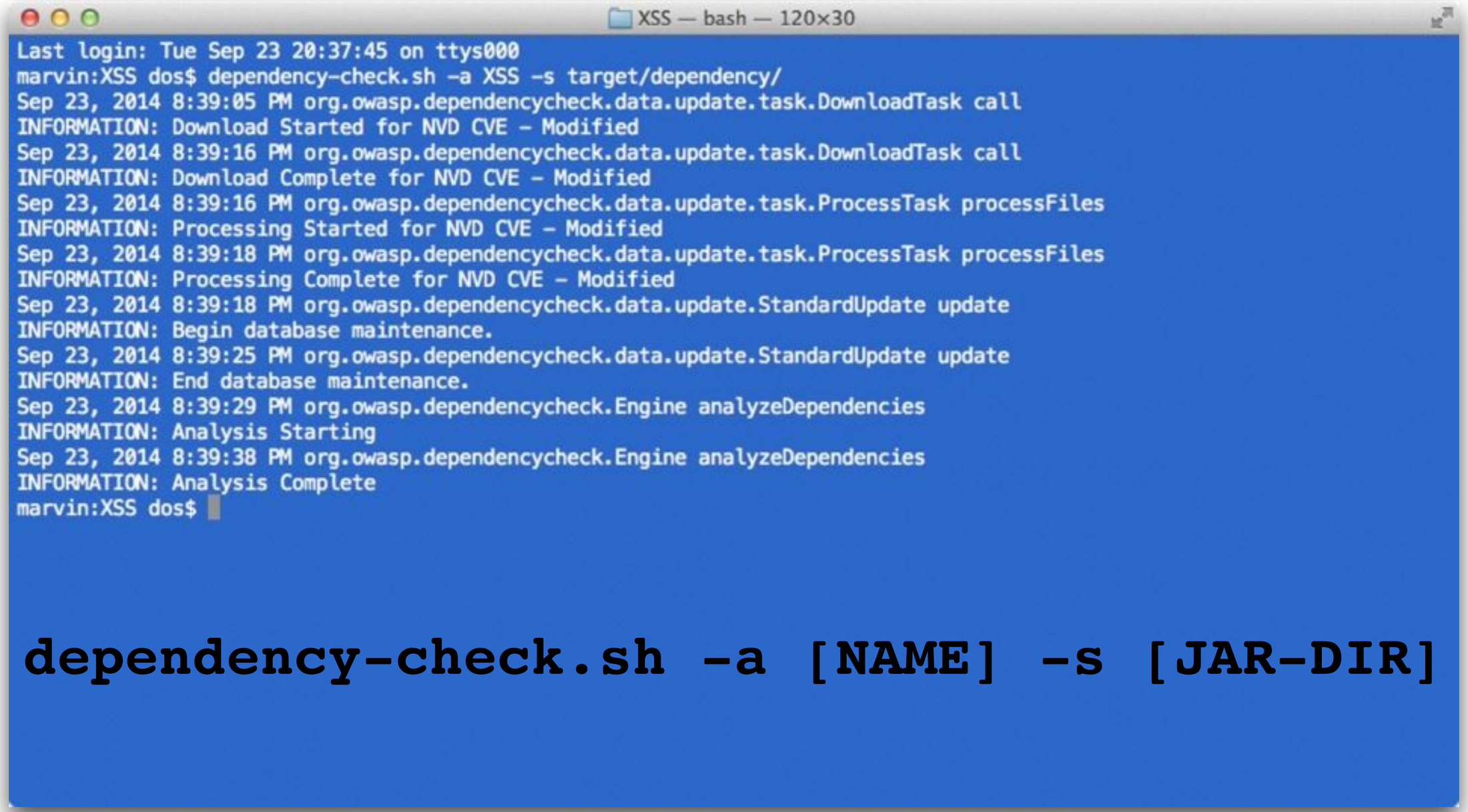
Keyczar uses JSON for key sets and keys

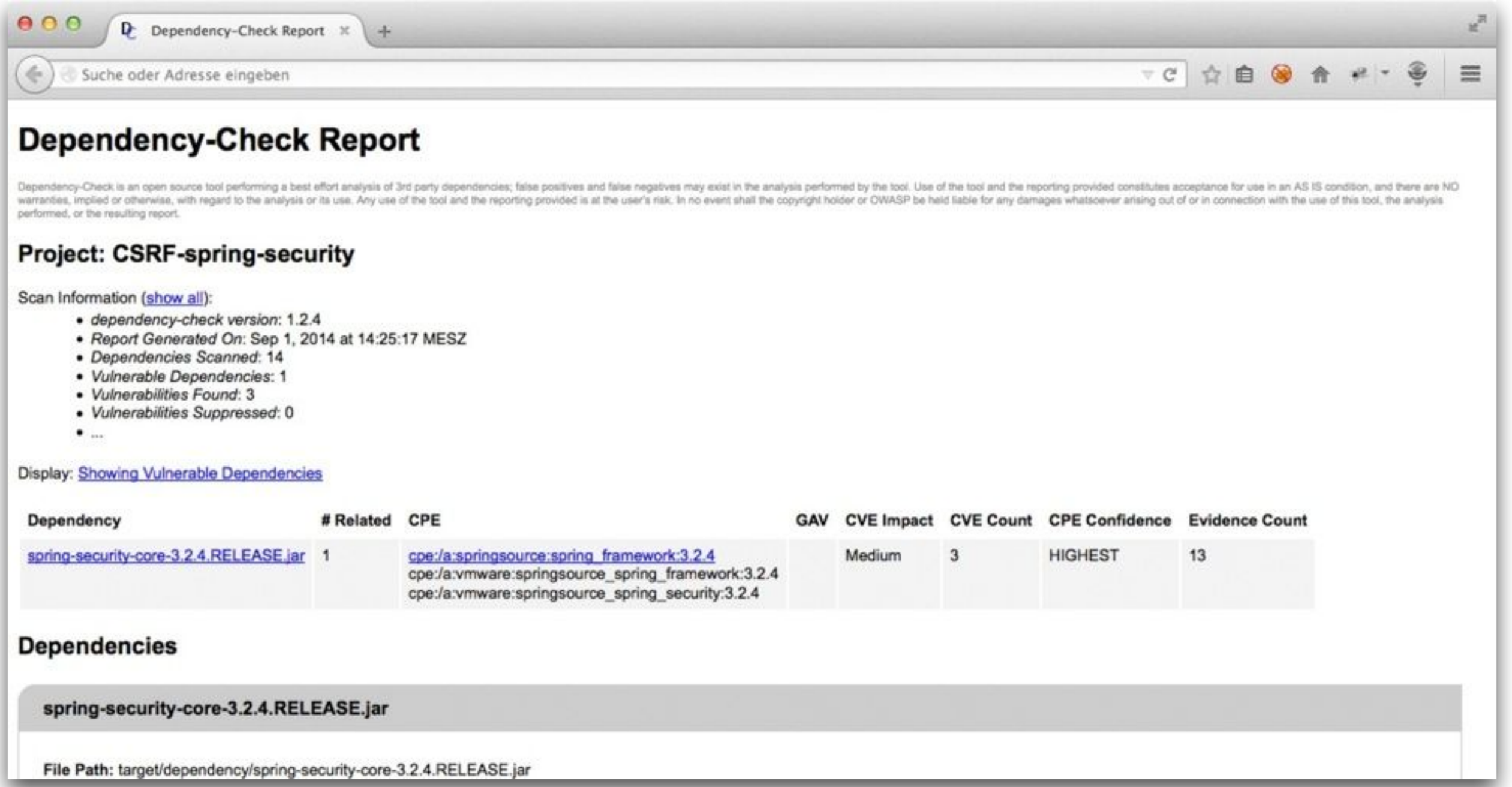
DEMO

Vulnerabilities in widespread libs affect many applications

Always keep your dependencies up-to-date

OWASP Dependency Check





Dependency-Check Report

Project: CSRF-spring-security

Scan Information (show all):

- *dependency-check version*: 1.2.4
- *Report Generated On*: Sep 1, 2014 at 14:25:17 MESZ
- *Dependencies Scanned*: 14
- *Vulnerable Dependencies*: 1
- *Vulnerabilities Found*: 3
- *Vulnerabilities Suppressed*: 0
- ...

Display: Showing Vulnerable Dependencies

Dependency	# Related	CPE	GAV	CVE Impact	CVE Count	CPE Confidence	Evidence Count
spring-security-core-3.2.4.RELEASE.jar	1	cpe:/a:springsource:spring_framework:3.2.4 cpe:/a:vmware:springsource_spring_framework:3.2.4 cpe:/a:vmware:springsource_spring_security:3.2.4		Medium	3	HIGHEST	13

Dependencies

spring-security-core-3.2.4.RELEASE.jar

File Path: target/dependency/spring-security-core-3.2.4.RELEASE.jar

Use Dependency Check as Jenkins plug-in

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

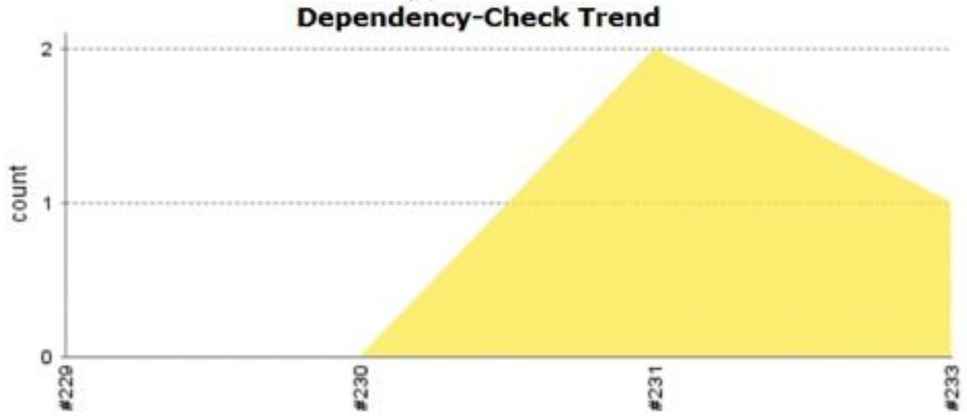
Details

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



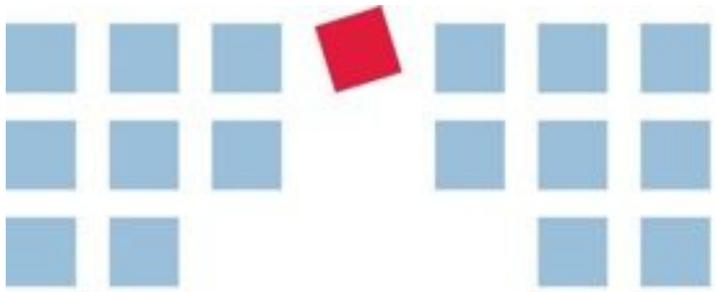
OWASP Dependency Check features

Known vulnerabilities check (publicly disclosed)

- Automatic update of vulnerability information
- Execution via
  - Command line
  - Ant task
  - Maven plug-in
  - Jenkins plug-in

Restrict yourself on (security) library usage

- Don’t reinvent the wheel, use existing libraries
- Keep libs up-to-date
- Test and hack your own applications



bridging IT

BridgingIT GmbH
Koenigstr. 42
70173 Stuttgart/ Germany

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

Blog blog.dominikschadow.de
Twitter @dschadow

Demo Projects
github.com/dschadow/JavaSecurity

Microsoft Threat Modeling Tool
www.microsoft.com/en-us/download/details.aspx?id=42518

Recx Security Analyser
chrome.google.com/webstore/detail/recx-security-analyser/ljafjhbjenhgcgnikniijchkngljgjda

Firebug
addons.mozilla.org/en-US/firefox/addon/firebug

OWASP Zed Attack Proxy
www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

FoxyProxy
addons.mozilla.org/de/firefox/addon/foxyproxy-standard

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

Jasypt
www.jasypt.org

Keyczar
www.keyczar.org

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

Spring Security
projects.spring.io/spring-security

Apache Shiro
shiro.apache.org

Pictures
www.dreamstime.com