



Security-by-Design Durch Einsatz von MVC

SecureNet GmbH

OWASP

Frankfurt, 25.11.08

Mirko Richter

Matthias Rohr

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

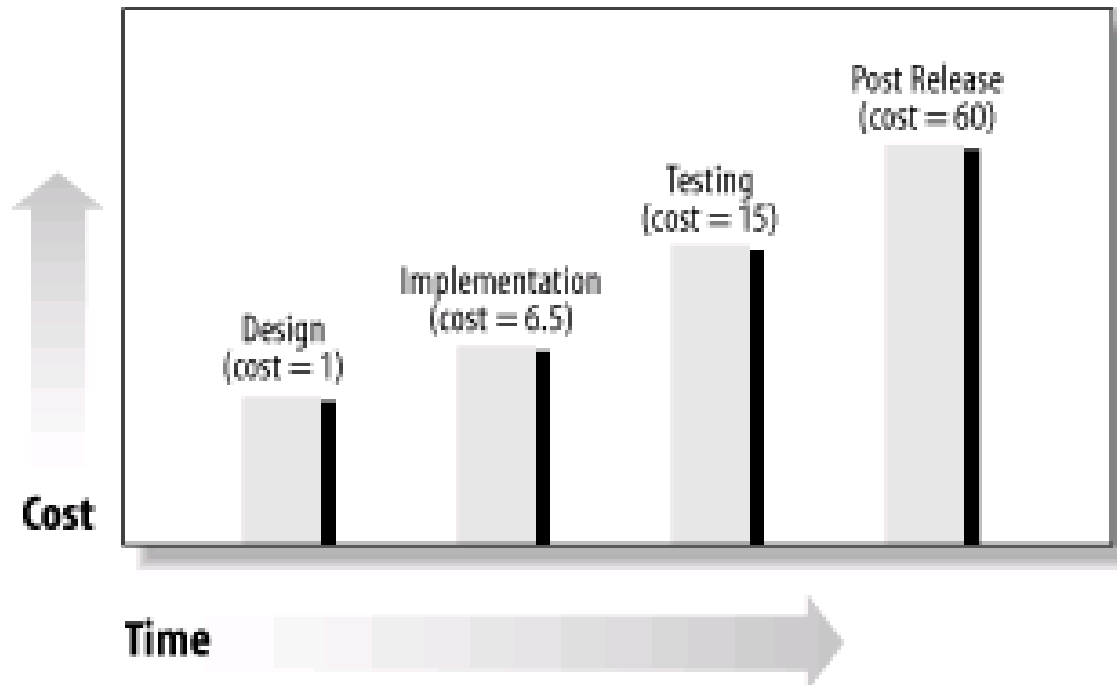
<http://www.owasp.org>

Agenda

- Security-by-Design
- MVC – Model-View-Controller
- Sicherheit durch MVC
- Probleme von MVC
- Sicherheit durch Erweiterungen
- Zusammenfassung
- Literatur

Security-by-Design

- Schwachstellen entstehen meist auf Implementierungsebene
- Ziel: Schwachstellen so früh wie möglich adressieren
- Kostenersparnis bis zu Faktor 60!

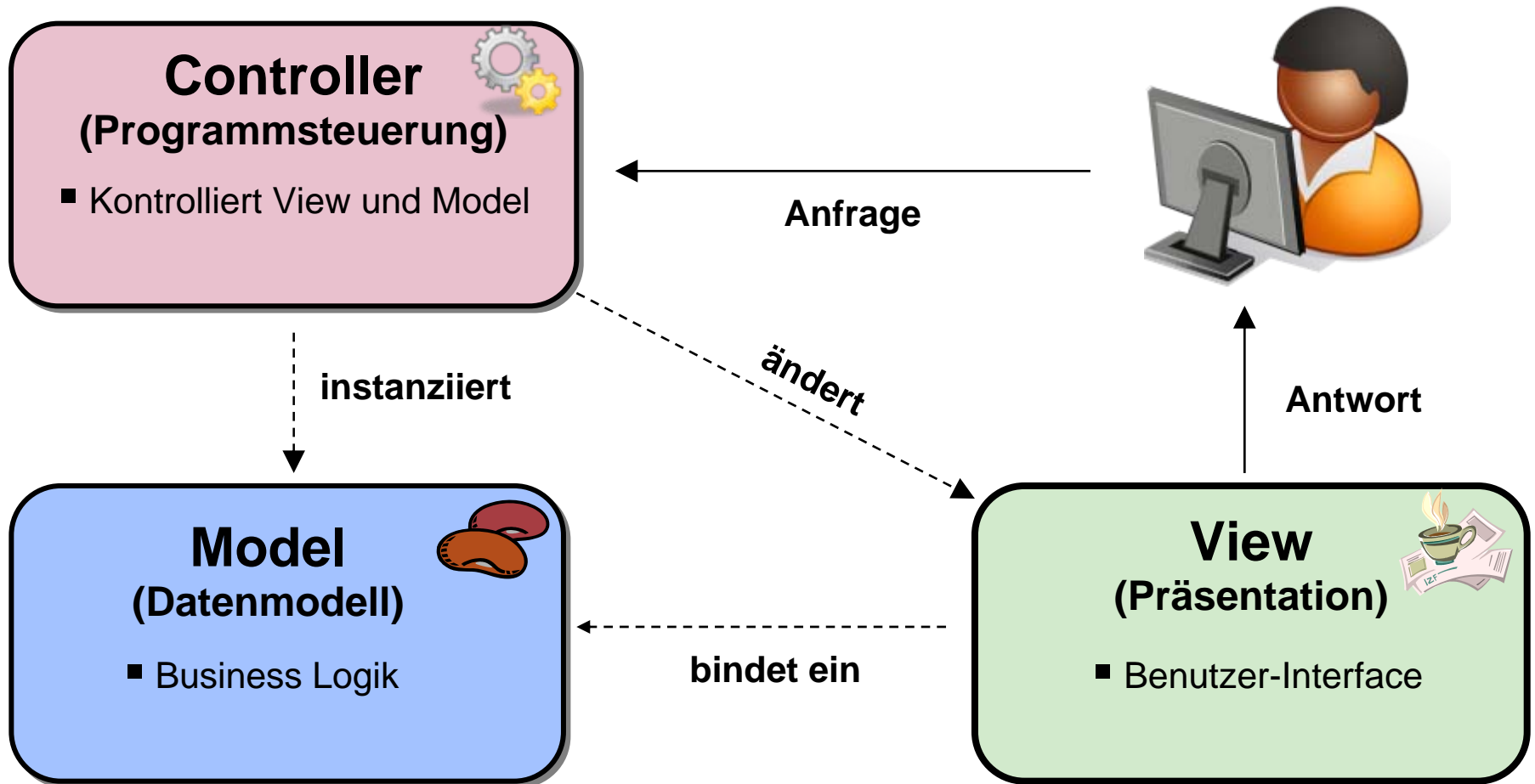


Aus: Secure Coding Principley & Practices

MVC – Was ist das?

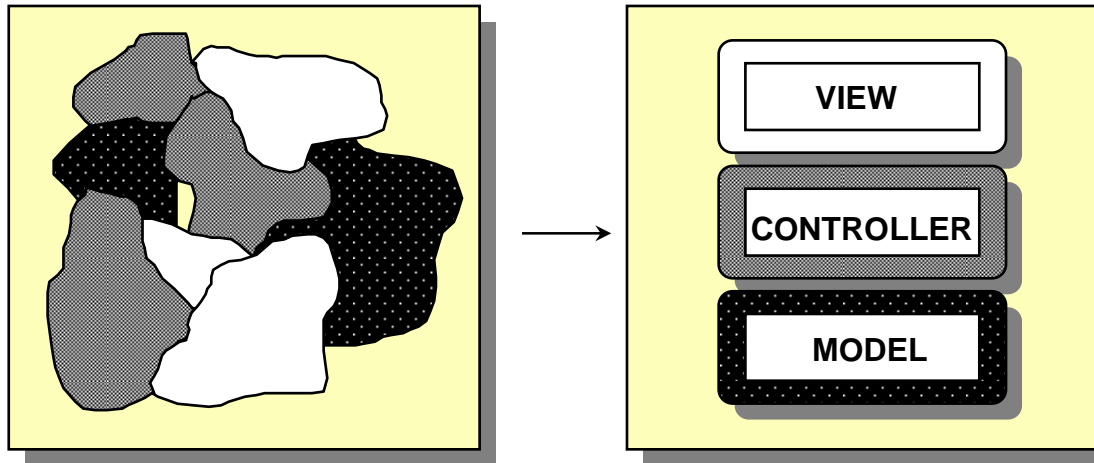
- Model / View / Controller
- Ursprung im Smalltalk (1970er Jahre)
- Architekturmuster zur Strukturierung von Software in 3 Ebenen:
 - Datenmodell (M)
 - Präsentation (V)
 - Programmsteuerung (C)

MVC - Übersicht



MVC – Warum?

- Unterteilt die Anwendung nach Verantwortlichkeiten
- Verbessert:
 - Wiederverwendbarkeit
 - Verständlichkeit
 - Anpassbarkeit
 - *Sicherheit*

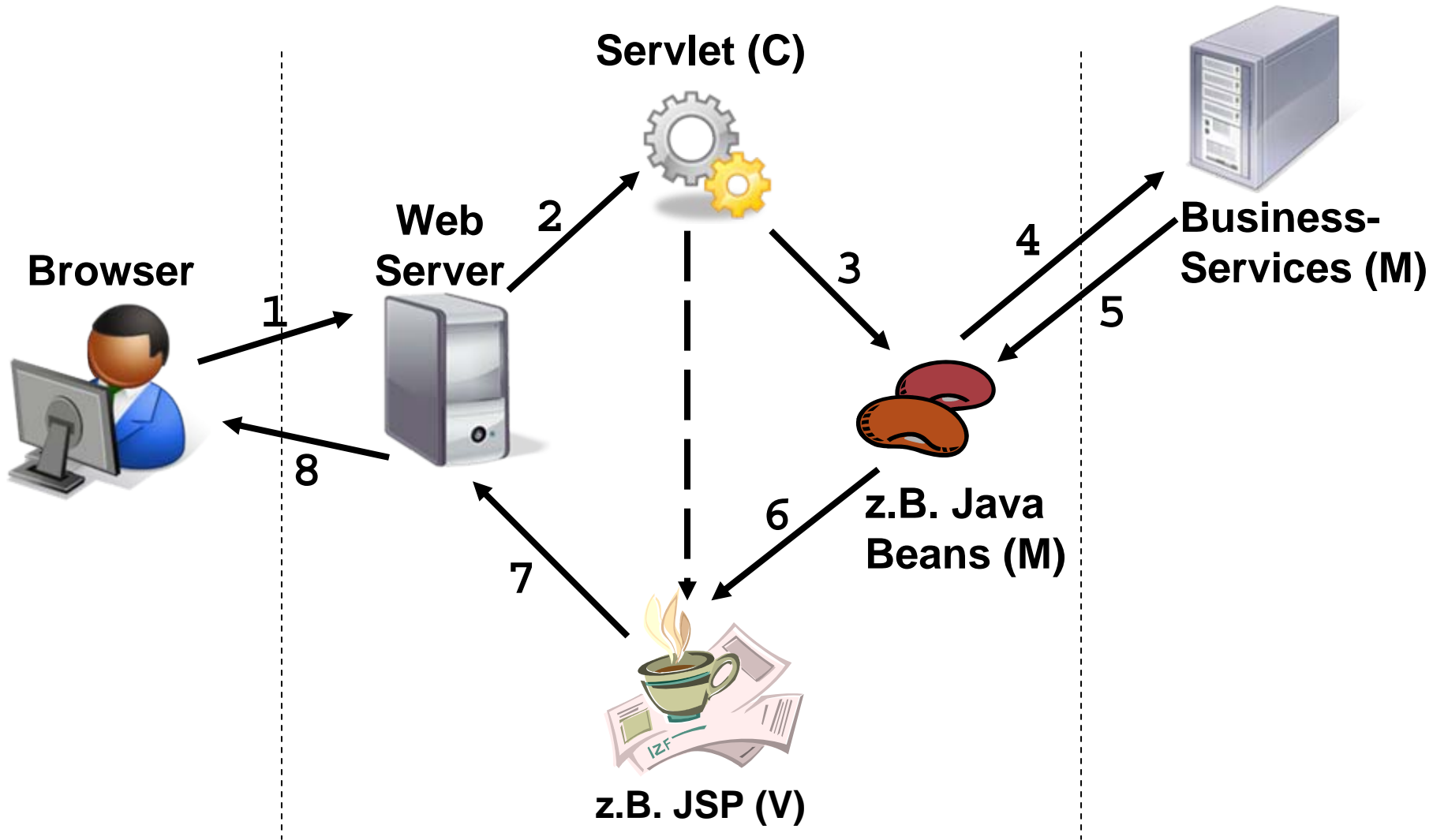


MVC – Web-Frameworks

- Java
 - Struts
 - **Spring MVC**
 - JSF
 - JBoss Seam
- ASP.Net
- Ruby on Rails
- PHP
 - Symfony
 - Cake PHP
- [...]



MVC – Web-Applikation



OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

Quelle: http://www.owasp.org/index.php/Top_10_2007

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Missbrauch des Vertrauensverhältnisses vom Browser zum Serverinhalt (zusätzlicher ungewollter Inhalt)

➤ Session Hijacking

➤ Website Spoofing

➤ Fernsteuerung des Browsers

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Ausnutzung ungenügender Datenvalidierung

➤ SQL – Injection

➤ OS-Command Injection

➤ Code – Injection

➤ XPath - Injection

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Ausführung von unerwünschtem Code auf dem Server

➤ Remote File-Include (PHP)

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Zugriff auf Daten, die nicht für den aktuellen Nutzer gedacht sind

➤ Privilege Escalation

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Missbrauch des Vertrauensverhältnisses vom Server zur Bowseranfrage (Ausnutzung fremder Rechte)

➤ Session Riding

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Ausgabe von Informationen, die nicht für den Nutzer gedacht sind

➤ Information Disclosure

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ **Logische Fehler bei der Authentisierung und Autorisierung**

➤ **Passwort vergessen (eigene E-Mail)**

➤ **Logout (Browser-Back)**

➤ **Session Fixation**

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Sensitive Daten werden unverschlüsselt abgespeichert (z.B. Passwörter)

➤ Zugriff auf sensitive Daten

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ Daten können beim Transport „abgehört“ werden

➤ http vs. https

Quelle: http://www.owasp.org/index.php/Top_10_2007

OWASP Top Ten (2007)

A1 – Cross Site Scripting

A2 – Injection Flaws

A3 – Malicious File Execution

A4 – Insecure Direct Object Reference

A5 – Cross Site Request Forgery

A6 – Information Leakage & Improper Error Handling

A7 – Broken Authentication & Session Management

A8 – Insecure Cryptographic Storage

A9 – Insecure Communications

A10 – Failure to Restrict URL Access

➤ „Brute-Force“ auf die URL

➤ Path Traversal

➤ „Raten“ von URLs (z.B. /admin)

MVC – Sicherheitsrelevante Konzepte

Controller



Single Access Point

Zentrales Fehlermanagement

Eingabevalidierung

Model



Data Binding

Validierung

View



Ausgabevalidierung

Data-Binding

- Beliebige Verschachtelungstiefe
z.B. a.b.c.name=foo
- Automatische Typ-
umwandlung



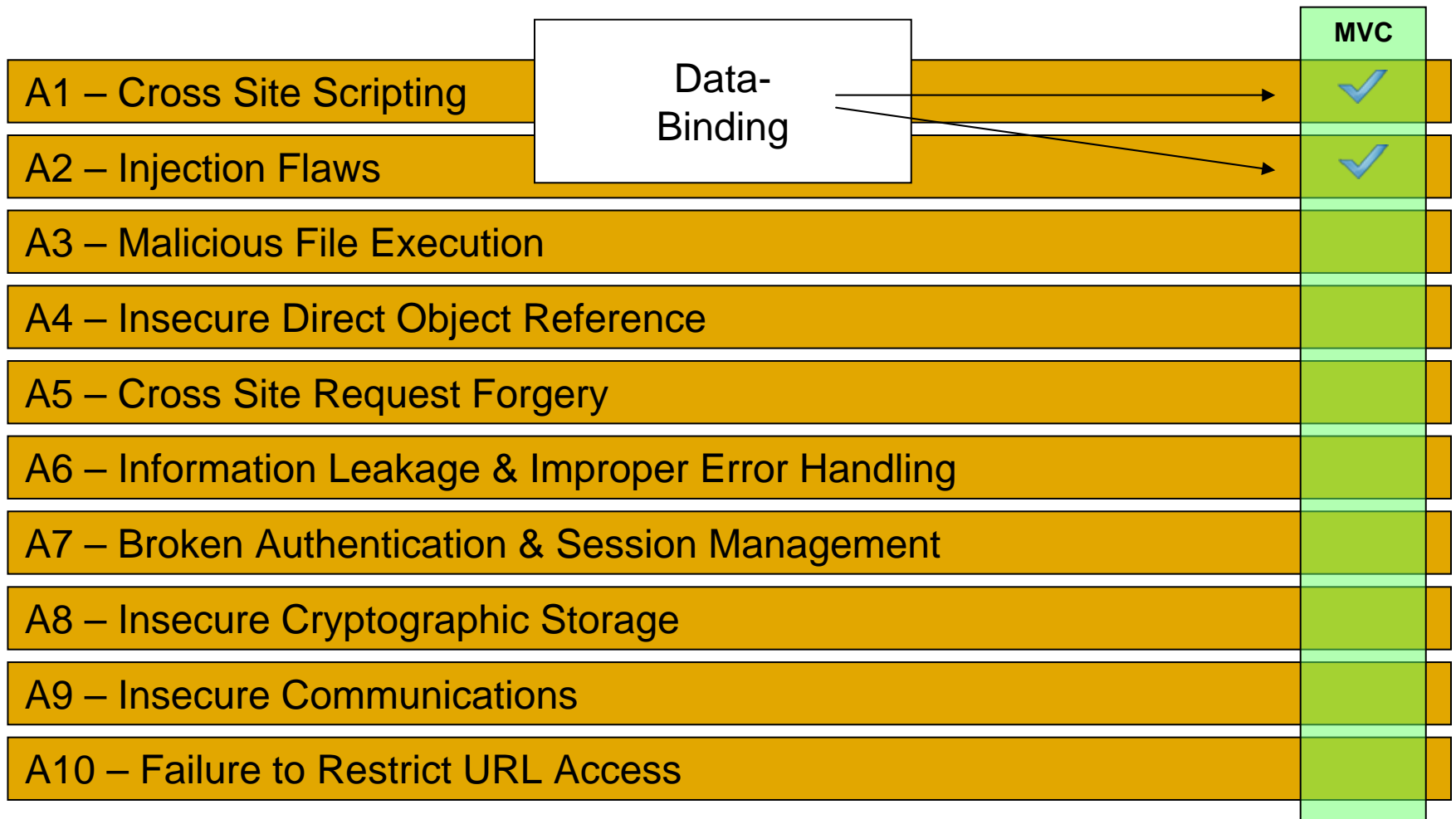
```
class Ob {  
    String streetName;  
    Integer streetNr;  
    public void setStreetName(String streetName) {this.streeName = streetName;}  
    public void setStreetNr(Integer streetNr) {this.streetNr = streeNr;}  
}
```

OWASP Top Ten (2007)

	MVC
A1 – Cross Site Scripting	
A2 – Injection Flaws	
A3 – Malicious File Execution	
A4 – Insecure Direct Object Reference	
A5 – Cross Site Request Forgery	
A6 – Information Leakage & Improper Error Handling	
A7 – Broken Authentication & Session Management	
A8 – Insecure Cryptographic Storage	
A9 – Insecure Communications	
A10 – Failure to Restrict URL Access	

Quelle: http://www.owasp.org/index.php/Top_10_2007

OWASP Top Ten (2007)



Quelle: http://www.owasp.org/index.php/Top_10_2007

Eingabevalidierung

- Mehrere Möglichkeiten im Spring-Framework
 - Spring Validatoren
 - Commons Validator
 - Valang
 - Bean Validation Framework

- Entscheidungskriterien
 - Programmatisch und/oder deklarativ (Code, Annotation, XML etc.)
 - Server- und/oder clientseitige Prüfung
 - Trennung der Verantwortlichkeiten (Domain-Model-Constraints, Data-Constraints etc.)

Eingabevalidierung mit Spring-Validatoren

```
public class MyValidator implements Validator
{
    public boolean supports(Class clazz)
    {
        return clazz.equals(MyCommand.class);
    }

    public void validate(Object obj, Errors errors)
    {
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "currency.ratio", "required");
        ValidationUtils.rejectIfEmptyOrWhitespace(errors, "currency.type", "required");
    }
}
```

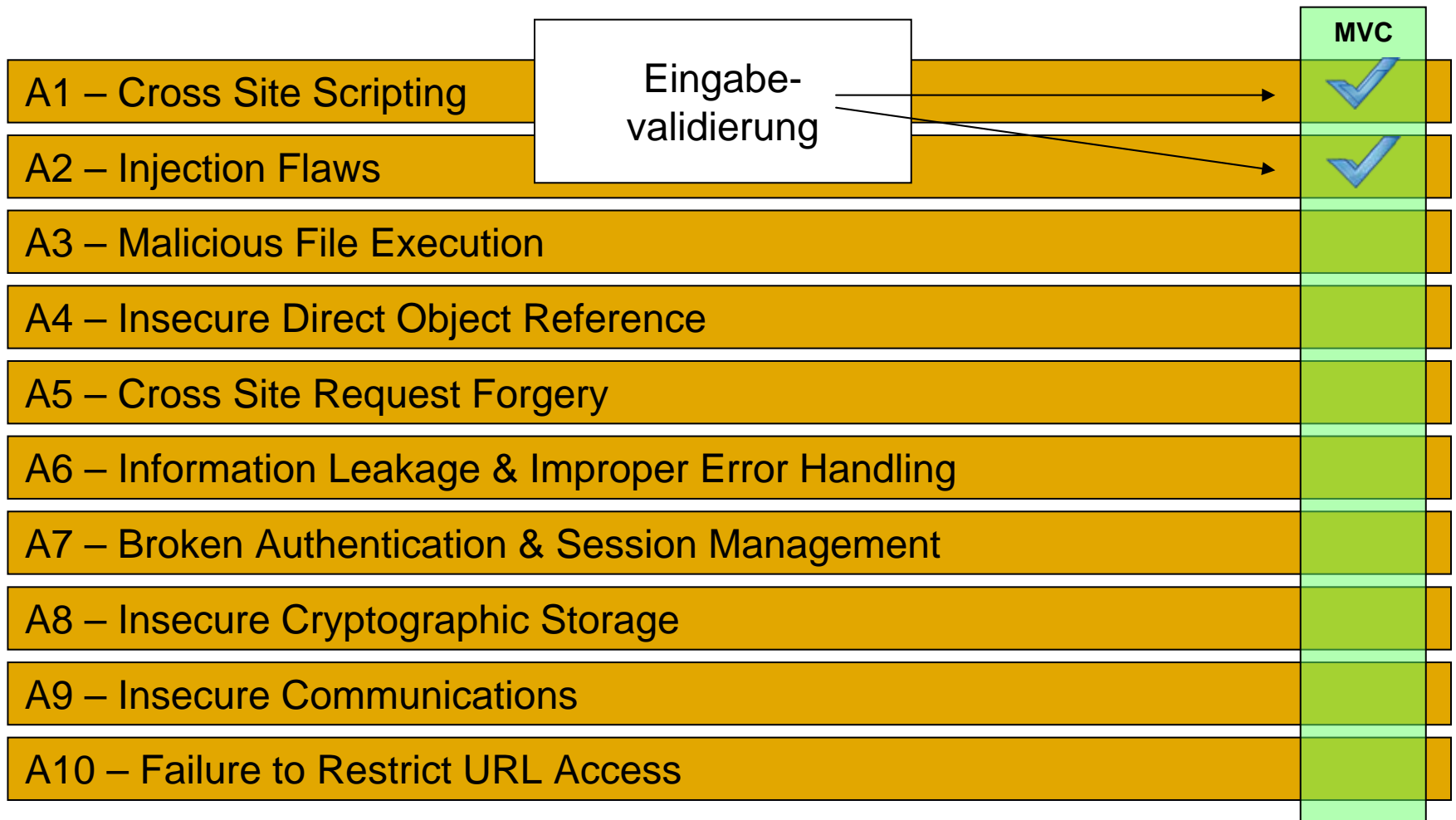
```
<bean name="/int/ca/list_supplier.html" class="..." autowire="byType">
    <property name="commandClass"><value>MyCommand</value></property>
    <property name="validator"><bean class="MyValidator"></bean></property>
</bean>
```

Eingabevalidierung mit *Valang*

- Deklarative Definition von Validierungsregeln
- Sowohl server- als auch client-seitig einsetzbar
- Einbindung des Spring-Modules erforderlich

```
<bean id="findUserValidator"  
      class="org.springframework.validation.ValangValidator">  
  <property name="valang">  
    <value><![CDATA[  
      { firstName : length(?) < 30 : 'First name too long' }  
      { lastName : length(?) < 50 : 'Last name too long' }  
    ]]></value>  
  </property>  
</bean>
```

OWASP Top Ten (2007)



Quelle: http://www.owasp.org/index.php/Top_10_2007

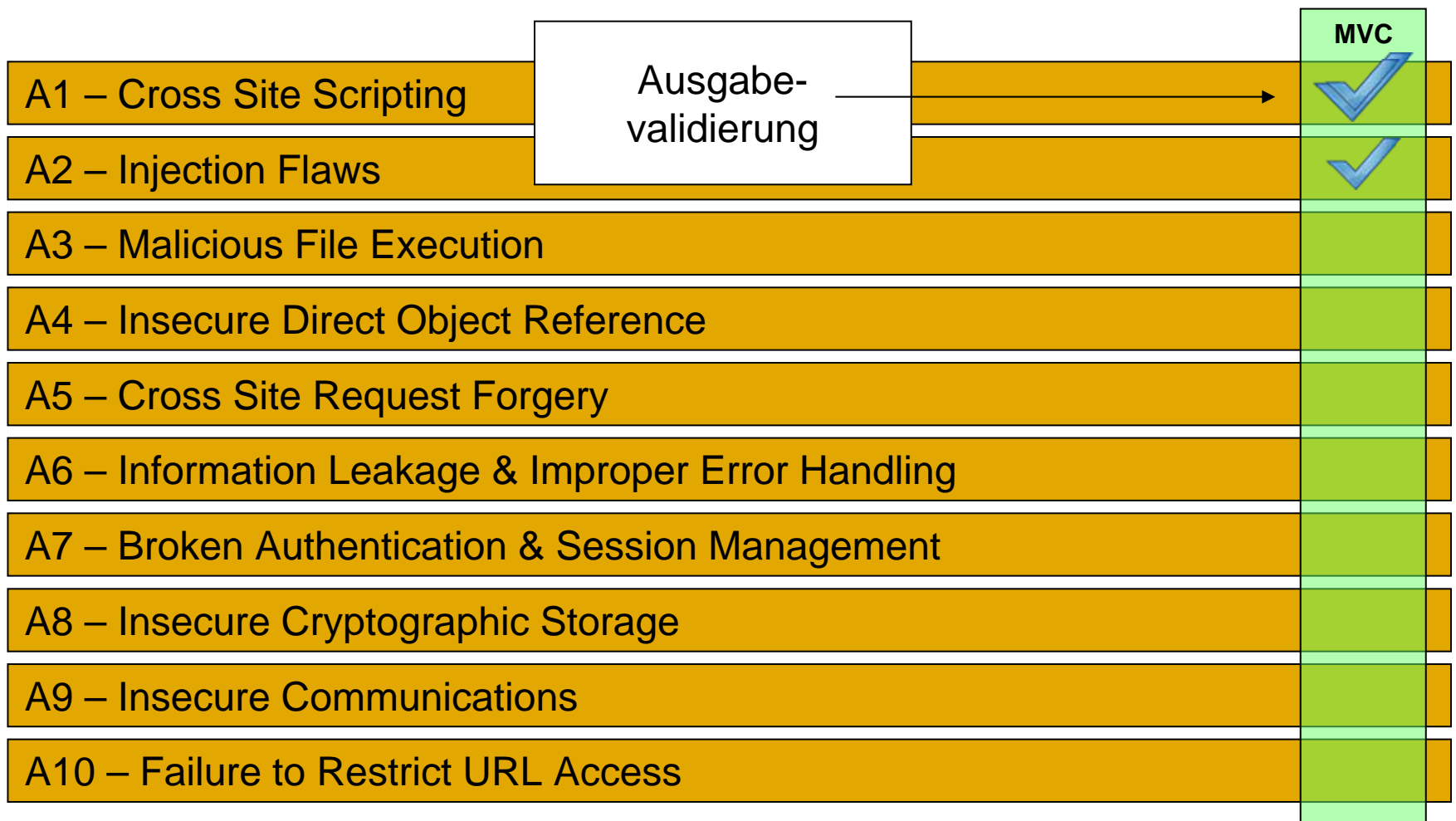
Ausgabevalidierung

- HTML-Escaping ist Standard bei den meisten JSTL-Tags

```
<c:set var="test" scope="page"><script>alert(0)</script></c:set>  
<c:out value="{test}" //> <!-- HTML Escaping >
```

- Verhindert viele Formen von Cross-Site-Scripting (XSS)
- Spring MVC VTL/FTL-Direktiven
 - Default: HTML-Encodierung aller Ausgaben
 - Parameter defaultHtmlEscape in web.xml
 - per-Tag definierbar, z.B. bei VTL: #set(\$springXhtmlCompliant = true)

OWASP Top Ten (2007)



Quelle: http://www.owasp.org/index.php/Top_10_2007

Zentrale Fehlerbehandlung

- Ausnahmen werden an definierter Stelle „oberhalb“ der Anwendung behandelt
- Default-Behandlung möglich und auch sinnvoll

```
<bean id="exceptionResolver" class="sample.ExceptionResolver">  
  <property name="exceptionMappings">  
    <props>  
      <prop key="UnauthorizedException">redirect:/...</prop>  
      <prop key="DataAccessException">redirect:/...</prop>  
    </props>  
  </property>  
  <property name="defaultErrorView"><value>redirect:/...</value></property>  
</bean>
```

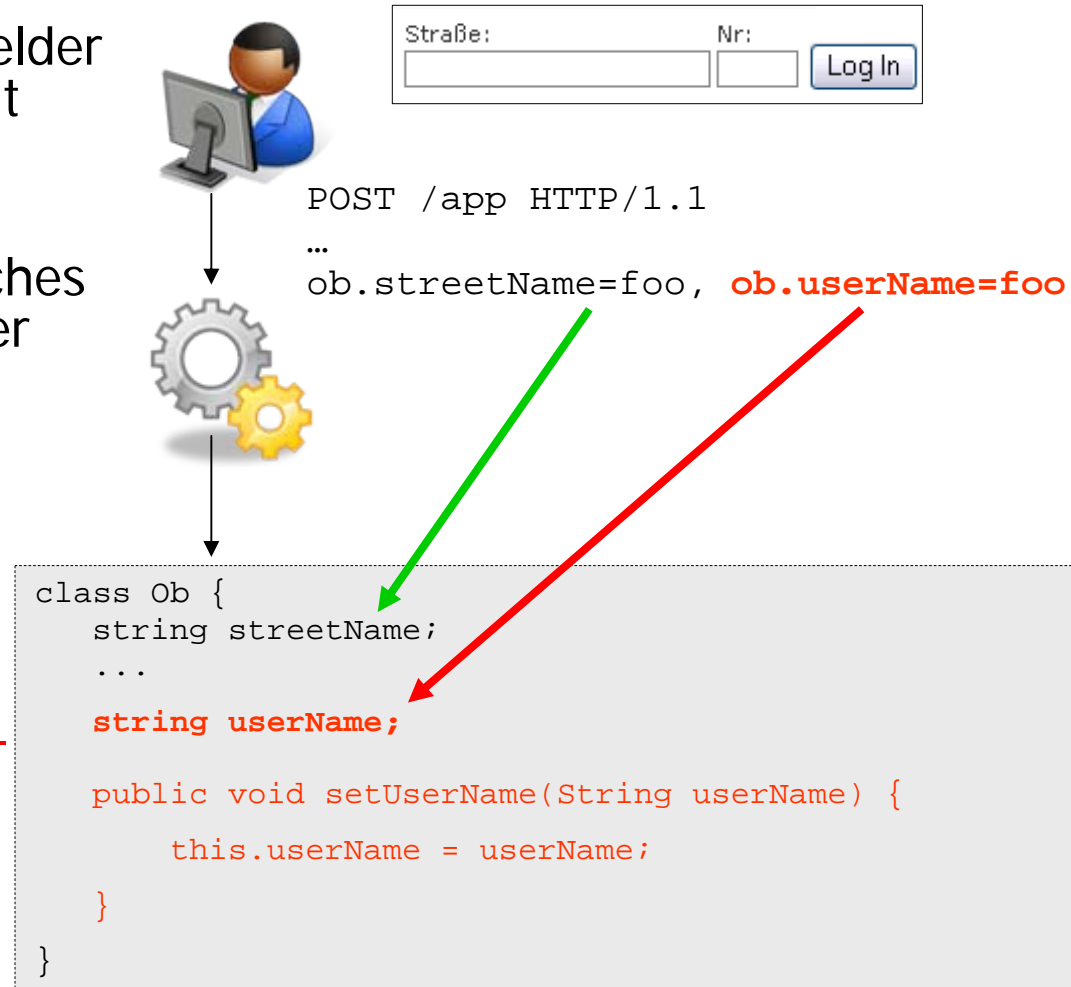
OWASP Top Ten (2007)

	zentrale Fehlerbehandlung	MVC
A1 – Cross Site Scripting		✓
A2 – Injection Flaws		✓
A3 – Malicious File Execution		
A4 – Insecure Direct Object Reference		
A5 – Cross Site Request Forgery		
A6 – Information Leakage & Improper Error Handling		✓
A7 – Broken Authentication & Session Management		
A8 – Insecure Cryptographic Storage		
A9 – Insecure Communications		
A10 – Failure to Restrict URL Access		

Quelle: http://www.owasp.org/index.php/Top_10_2007

Data Submission to Non-Editable Field

- JavaBeans können auch Felder haben, die der Nutzer nicht verändern soll
- Errät ein Angreifer ein solches „gebundenes“ Feld, kann er interne Variablen überschreiben ...
- ... und ggf. seine Privilegien erweitern etc.
- **Prinzipiell negative Auswirkung auf die meisten Schwachstellen vorstellbar**



Korrektur

- Konsequenz: Parameter, die fürs Binding erlaubt sind, müssen explizit gesetzt werden
- oder: entsprechend restriktives Datenmodell verwenden

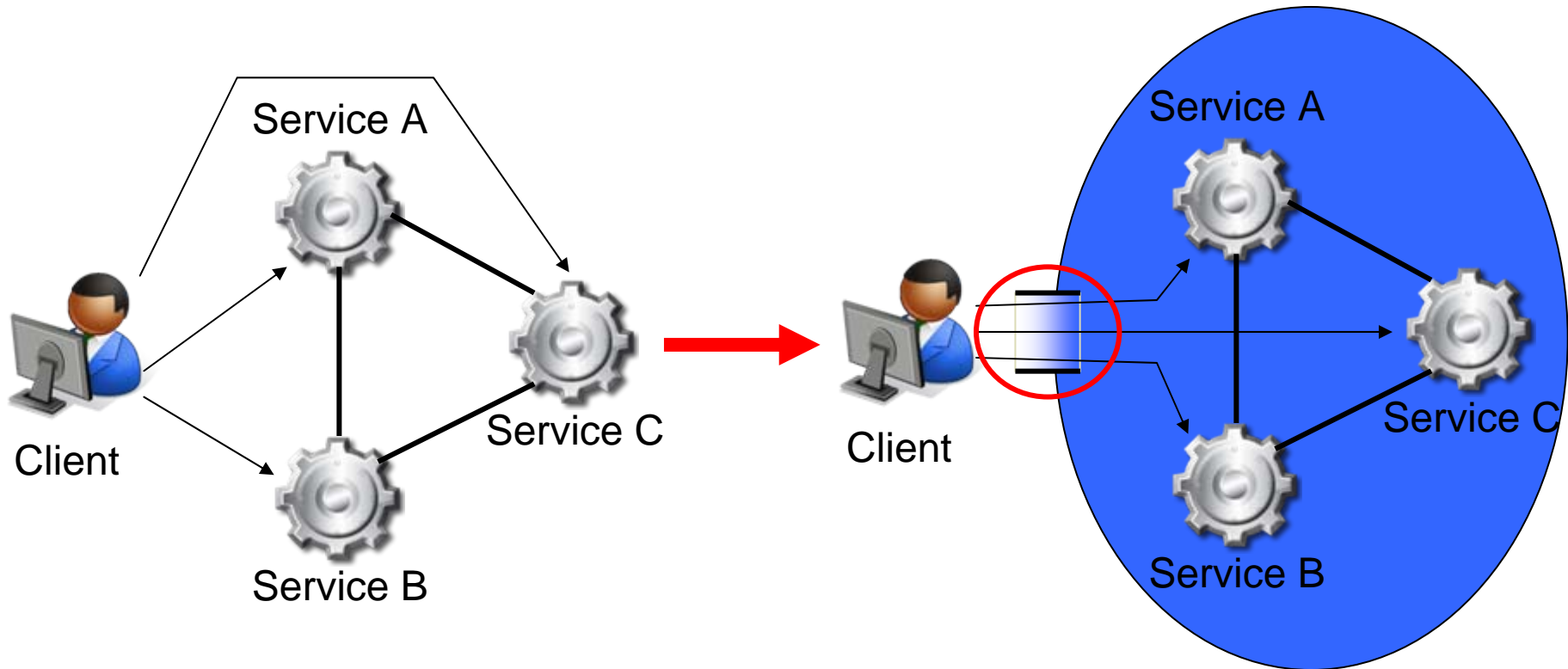
```
@Override
protected void initBinder(HttpServletRequest request,
ServletRequestDataBinder binder) throws Exception {
    binder.setAllowedFields(new String[] {"id", "name", "city"});
    binder.setRequiredFields(new String[] {"id"});
}
```

OWASP Top Ten (2007)

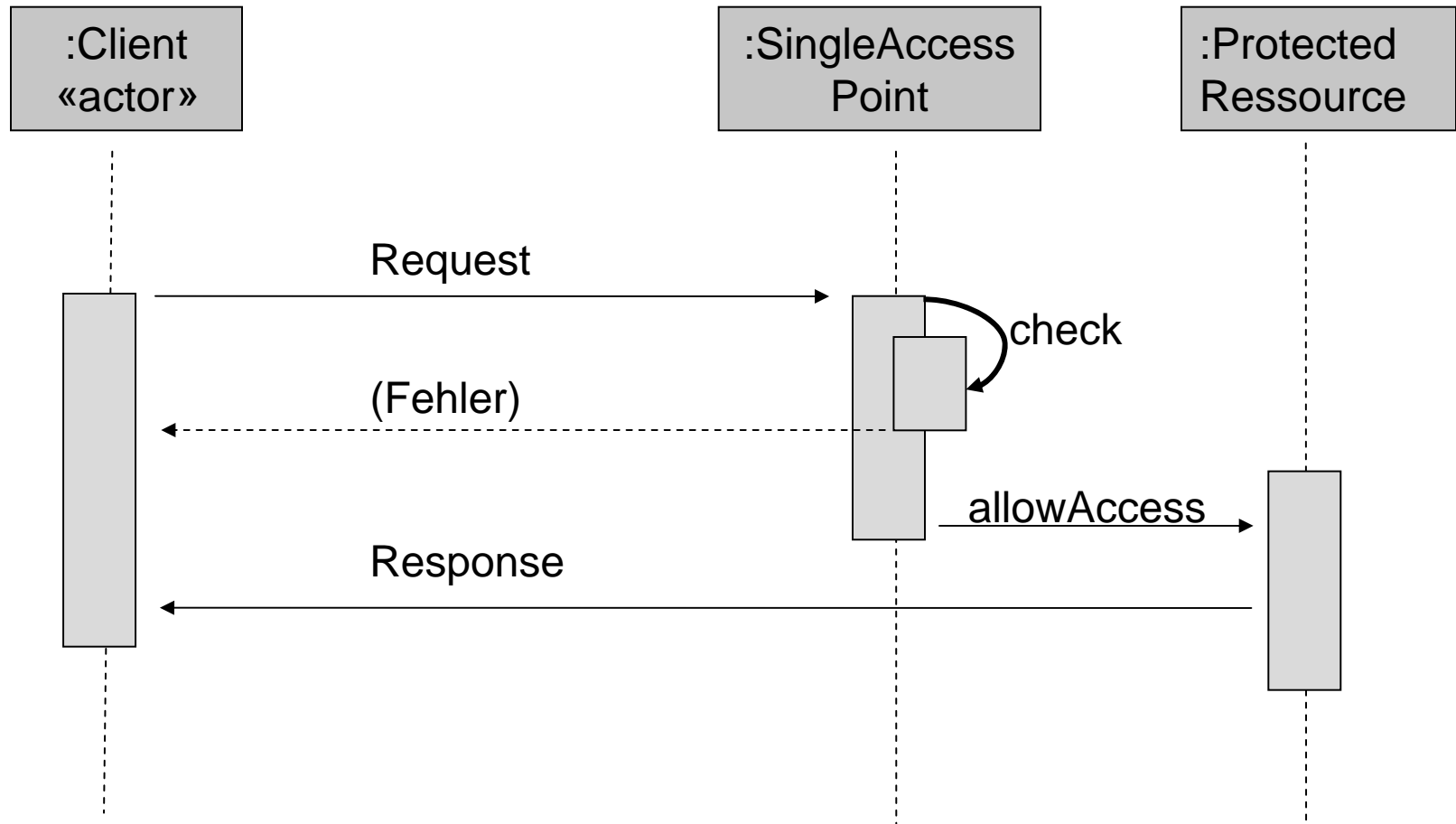
		MVC
A1 – Cross Site Scripting	Data Submission to Non-Editable Field	✓
A2 – Injection Flaws		✓
A3 – Malicious File Execution		
A4 – Insecure Direct Object Reference		
A5 – Cross Site Request Forgery		
A6 – Information Leakage & Improper Error Handling		✓
A7 – Broken Authentication & Session Management		
A8 – Insecure Cryptographic Storage		
A9 – Insecure Communications		
A10 – Failure to Restrict URL Access		

Quelle: http://www.owasp.org/index.php/Top_10_2007




Single Access Point (1)



Single Access Point (2)



OWASP Top Ten (2007)

		Erw. 1	MVC
A1 – Cross Site Scripting	Single Access Point		
A2 – Injection Flaws			
A3 – Malicious File Execution			
A4 – Insecure Direct Object Reference			
A5 – Cross Site Request Forgery			
A6 – Information Leakage & Improper Error Handling			
A7 – Broken Authentication & Session Management			
A8 – Insecure Cryptographic Storage			
A9 – Insecure Communications			
A10 – Failure to Restrict URL Access			

Quelle: http://www.owasp.org/index.php/Top_10_2007

Spring Security (1)

- Deklaratives Mapping von URL-basierten Berechtigungen

```
<http auto-config="true">
  // erlaube anonymen generell anonymen Zugriff
  <intercept-url pattern="/**" access="IS_AUTHENTICATED_ANONYMOUSLY" />

  // Administrations-Bereich dürfen nur für Admins
  <intercept-url pattern="/admin/**" access="ROLE_SUPERVISOR" />

  // post.html dürfen nur Benutzer mit Rolle ROLE_TELLER aufrufen
  <intercept-url pattern="/post.html" access="ROLE_USER" />
</http>
```

Spring Security (2)

➤ Globale Berechtigung durch Aspekte

```
<global-method-security secured-annotations="enabled">
  <protect-pointcut
    expression="execution(* bigbank.*Service.post*(..))"
    access="ROLE_ADMIN"/>
</global-method-security>
```

- Alle Service-Methoden die mit "post" beginnen dürfen nur von Rolle "ROLE_ADMIN" aufgerufen werden
- Auch über Annotationen möglich (z.B. "@SECURED ({ROLE_ADMIN})")
 - Programmatische Nähe zum Entwickler
 - Verbesserte Lesbarkeit & Wartbarkeit

OWASP Top Ten (2007)

		Erw. 1	MVC
A1 – Cross Site Scripting	URL-basierte Berechtigung		✓
A2 – Injection Flaws			✓
A3 – Malicious File Execution			
A4 – Insecure Direct Object Reference			
A5 – Cross Site Request Forgery			
A6 – Information Leakage & Improper Error Handling			✓
A7 – Broken Authentication & Session Management			
A8 – Insecure Cryptographic Storage			
A9 – Insecure Communications			
A10 – Failure to Restrict URL Access		✓*	

* Spring Security

Quelle: http://www.owasp.org/index.php/Top_10_2007

OWASP Top Ten (2007)

		Erw. 1	MVC
A1 – Cross Site Scripting	Aspekt-basierte Berechtigung		✓
A2 – Injection Flaws			✓
A3 – Malicious File Execution			
A4 – Insecure Direct Object Reference		✓*	
A5 – Cross Site Request Forgery			
A6 – Information Leakage & Improper Error Handling			✓
A7 – Broken Authentication & Session Management			
A8 – Insecure Cryptographic Storage			
A9 – Insecure Communications			
A10 – Failure to Restrict URL Access		✓*	

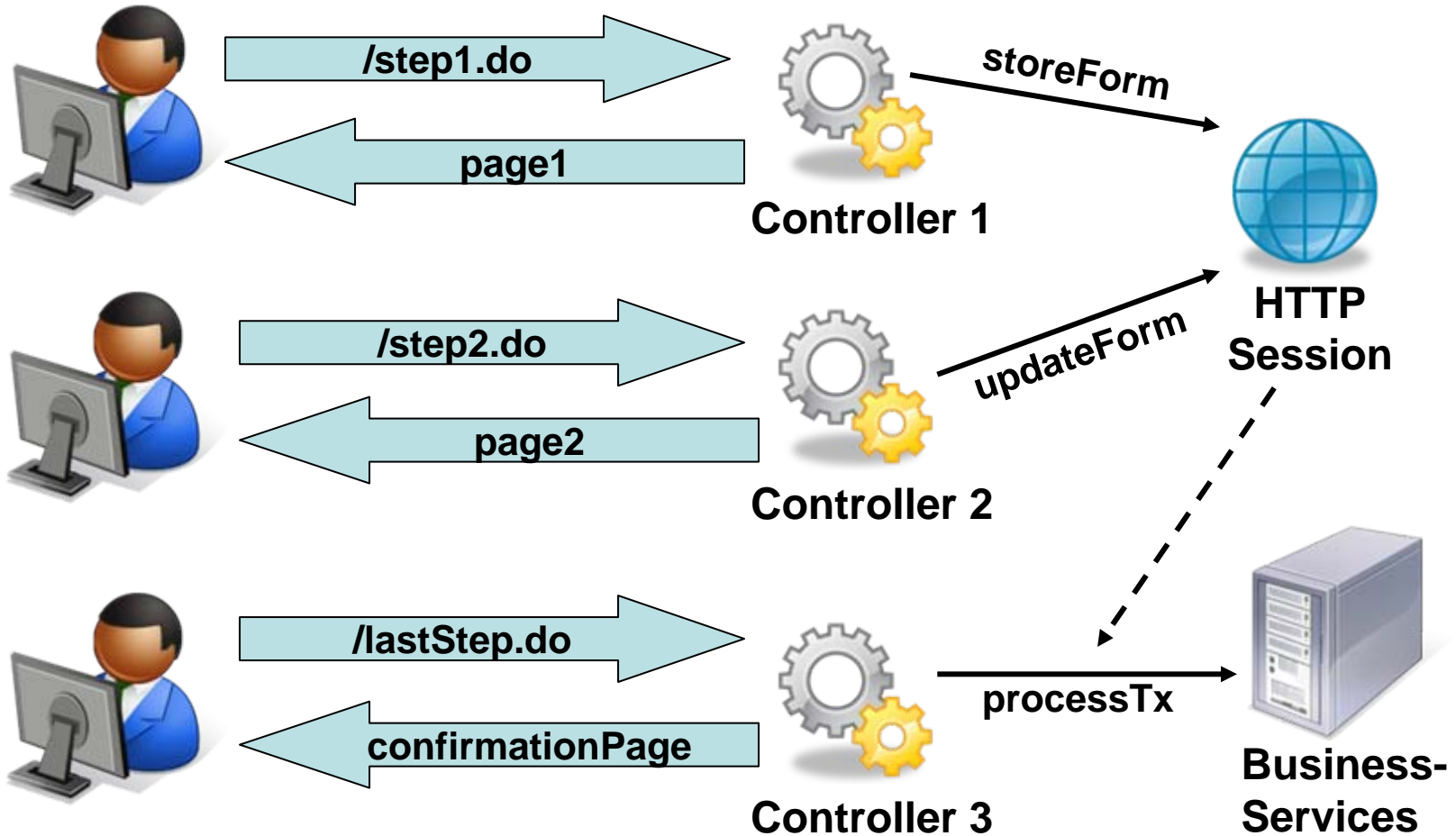
* Spring Security

Quelle: http://www.owasp.org/index.php/Top_10_2007

Web Flows (1)

- Häufig werden bestimmte Dialogabfolgen angenommen
- Shop-Beispiel:
 - Produkt in Einkaufswagen legen
 - Zur Kasse gehen
 - Rechnungs- & Lieferdaten eingeben
 - Bezahlen
 - Fertig
- Angreifer sind nicht an diesen Ablauf gebunden (Forceful Browsing)
- Anwendung kann so ggf. in unsicheren State versetzt werden.
- Mit Web Flows lassen sich Dialogabläufe erzwingen

Web Flows (2)



OWASP Top Ten (2007)

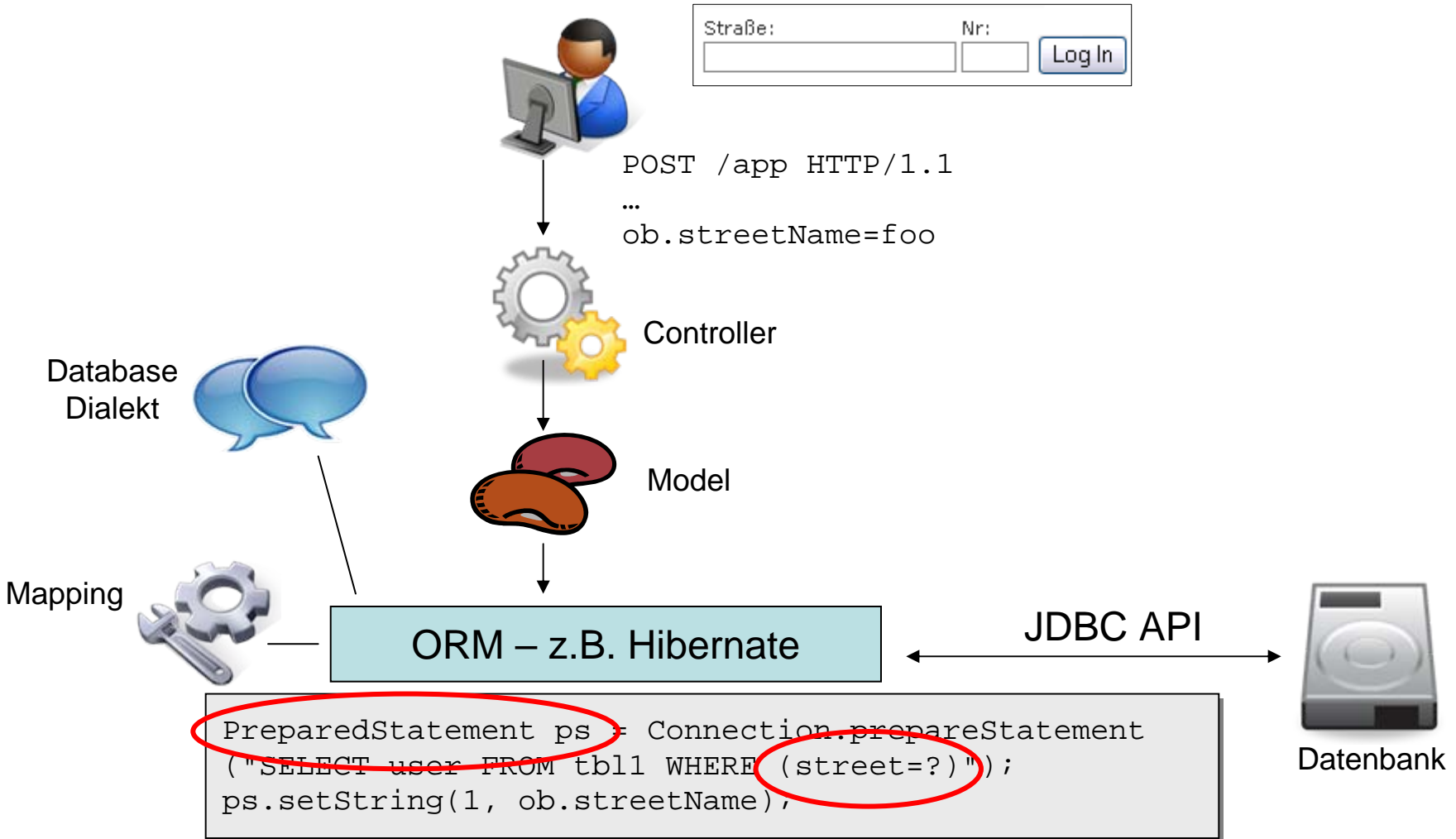
	Web-Flows	Erw. 1	MVC
A1 – Cross Site Scripting			✓
A2 – Injection Flaws			✓
A3 – Malicious File Execution			
A4 – Insecure Direct Object Reference		✓*	
A5 – Cross Site Request Forgery		✓**	
A6 – Information Leakage & Improper Error Handling			✓
A7 – Broken Authentication & Session Management			
A8 – Insecure Cryptographic Storage			
A9 – Insecure Communications			
A10 – Failure to Restrict URL Access		✓*	

* Spring Security

** Spring Web Flow

Quelle: http://www.owasp.org/index.php/Top_10_2007

ORM – Object Relational Mapping



OWASP Top Ten (2007)

	Erw. 1	MVC
A1 – Cross Site Scripting		✓
A2 – Injection Flaws	✓ ^{***}	✓
A3 – Malicious File Execution		
A4 – Insecure Direct Object Reference	✓ [*]	
A5 – Cross Site Request Forgery	✓ ^{**}	
A6 – Information Leakage & Improper Error Handling		✓
A7 – Broken Authentication & Session Management		
A8 – Insecure Cryptographic Storage		
A9 – Insecure Communications		
A10 – Failure to Restrict URL Access	✓ [*]	

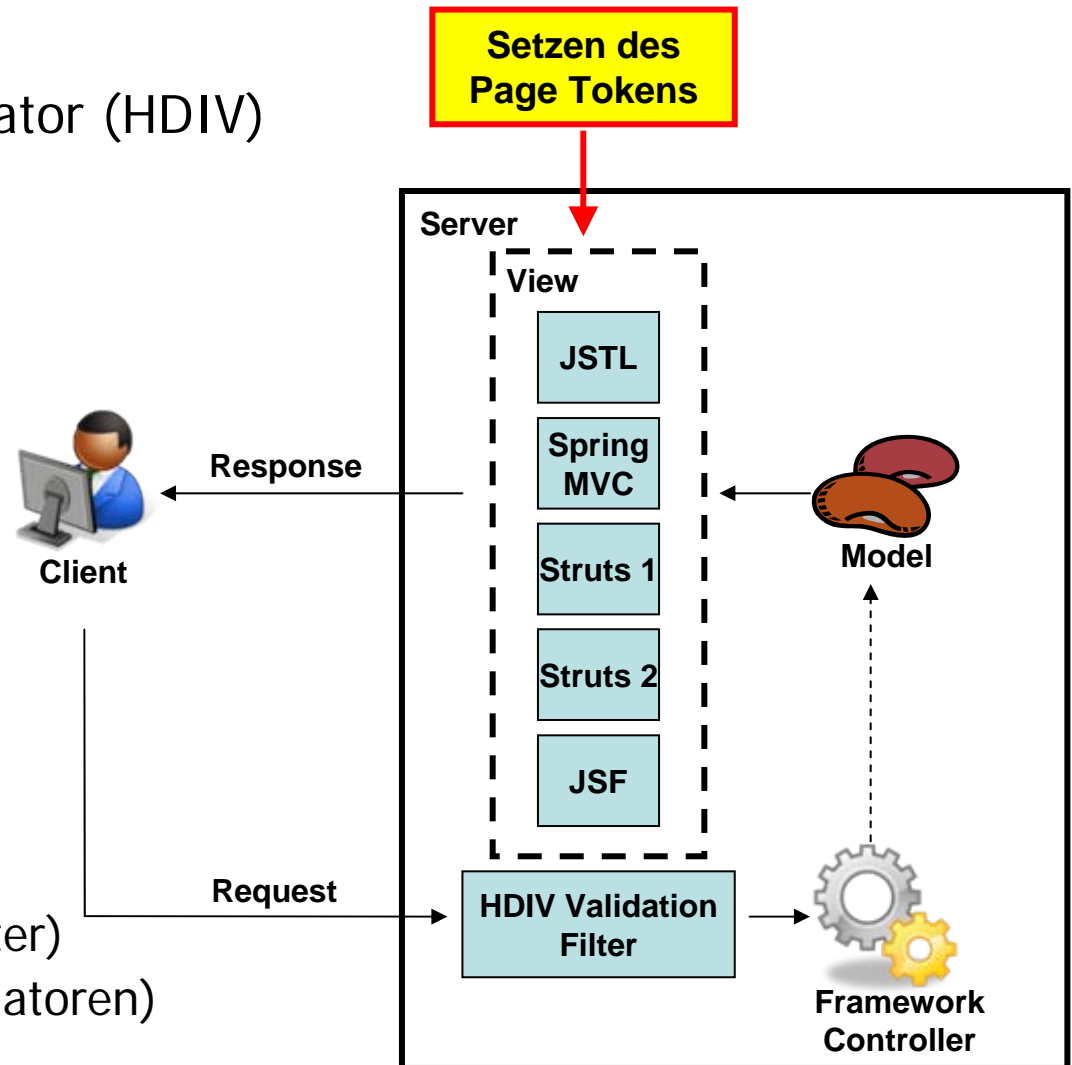
Prepared Statements

- * Spring Security
- ** Spring Web Flow
- *** ORM (Hibernate)

Quelle: http://www.owasp.org/index.php/Top_10_2007

HDIV

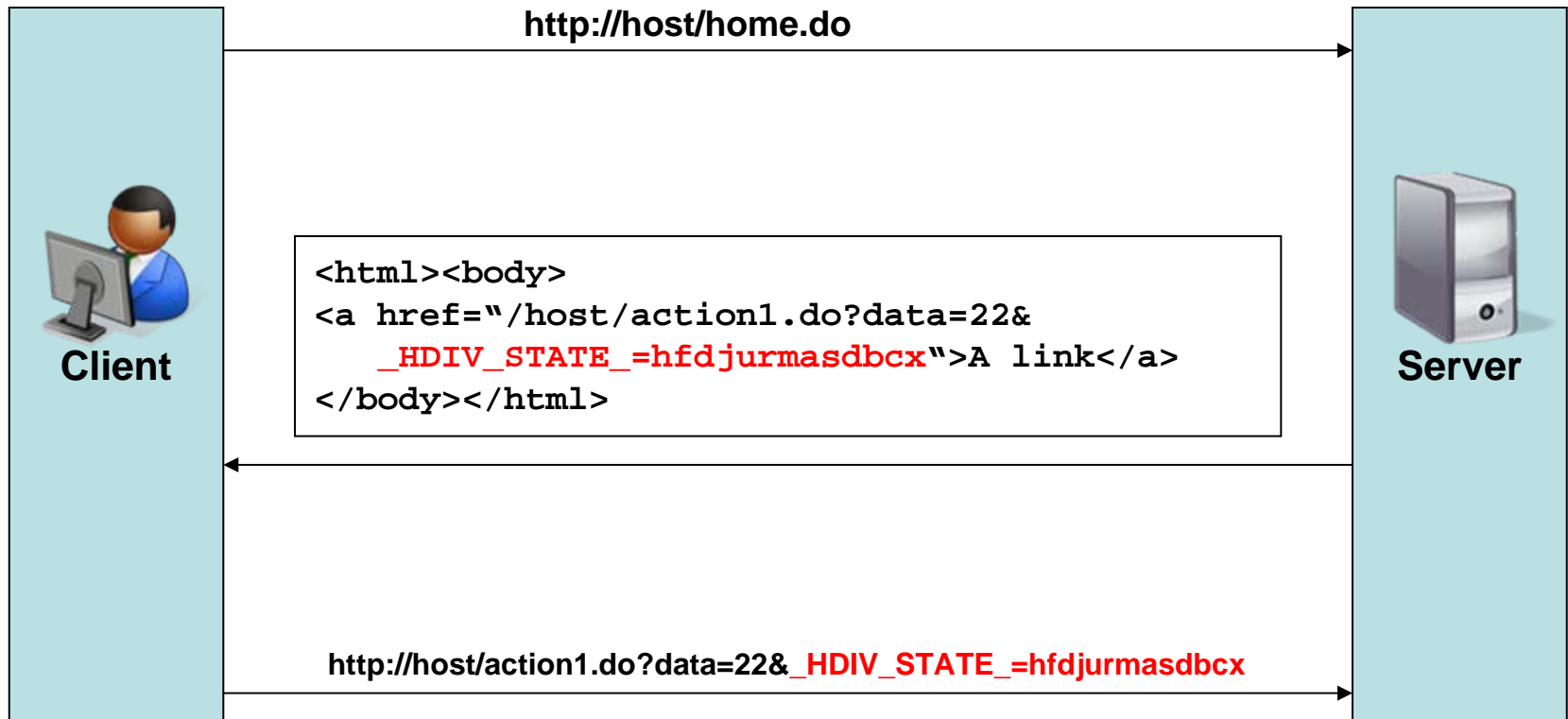
- HTTP Data Integrity Validator (HDIV)
- Open Source Projekt
- Unterstützt:
 - Struts (1.x und 2.x)
 - Spring MVC (ab 2.x)
 - JSTL
- Komponenten
 - Tag Libraries (Output Filter)
 - Servlet Filter (Input Validatoren)



HDIV - Bestandteile

- Eingabevalidierung (ähnlich OWASP Stinger)
- Schutz vor Session Riding / CSRF
 - mittels Page Tokens
- Schutz vor Forceful Browsing
 - mittels Session States
- Schutz vor Parametermanipulation ("Smart Form Protection")
 - Editierbar (<input>, <textarea>, ...)
 - Nicht-Editierbar (alle übrigen, z.B. Hidden Fields)

HDIV - Funktionsweise



OWASP Top Ten (2007)

	HDIV Eingabevalidierung	Erw. 2	Erw. 1	MVC
A1 – Cross Site Scripting	→	✓ +		✓
A2 – Injection Flaws	→	✓ +	✓ ***	✓
A3 – Malicious File Execution				
A4 – Insecure Direct Object Reference			✓ *	
A5 – Cross Site Request Forgery			✓ **	
A6 – Information Leakage & Improper Error Handling				✓
A7 – Broken Authentication & Session Management				
A8 – Insecure Cryptographic Storage				
A9 – Insecure Communications				
A10 – Failure to Restrict URL Access			✓ *	

* Spring Security + HDIV
 ** Spring Web Flow
 *** ORM (Hibernate)

Quelle: http://www.owasp.org/index.php/Top_10_2007

OWASP Top Ten (2007)

		Erw. 2	Erw. 1	MVC
A1 – Cross Site Scripting	HDIV Page Tokens	✓ +		✓
A2 – Injection Flaws		✓ +	✓ ***	✓
A3 – Malicious File Execution				
A4 – Insecure Direct Object Reference			✓ *	
A5 – Cross Site Request Forgery		✓ +	✓ **	
A6 – Information Leakage & Improper Error Handling				✓
A7 – Broken Authentication & Session Management				
A8 – Insecure Cryptographic Storage				
A9 – Insecure Communications				
A10 – Failure to Restrict URL Access			✓ *	

- * Spring Security + HDIV
- ** Spring Web Flow
- *** ORM (Hibernate)

Quelle: http://www.owasp.org/index.php/Top_10_2007

OWASP Top Ten (2007)

	Erw. 2	Erw. 1	MVC
A1 – Cross Site Scripting	✓ +		✓
A2 – Injection Flaws	✓ +	✓ ***	✓
A3 – Malicious File Execution			
A4 – Insecure Direct Object Reference	✓ +	✓ *	
A5 – Cross Site Request Forgery	✓ +	✓ **	
A6 – Information Leakage & Improper Error Handling			✓
A7 – Broken Authentication & Session Management			
A8 – Insecure Cryptographic Storage			
A9 – Insecure Communications			
A10 – Failure to Restrict URL Access	✓ +	✓ *	

HDIV
Session States

- * Spring Security + HDIV
- ** Spring Web Flow
- *** ORM (Hibernate)

Quelle: http://www.owasp.org/index.php/Top_10_2007

OWASP Top Ten (2007)

	Smart Form Protection	Erw. 2	Erw. 1	MVC
A1 – Cross Site Scripting	Smart Form Protection	✓ +		✓
A2 – Injection Flaws		✓ +	✓ ***	✓
A3 – Malicious File Execution				
A4 – Insecure Direct Object Reference		✓ +	✓ *	
A5 – Cross Site Request Forgery		✓ +	✓ **	
A6 – Information Leakage & Improper Error Handling				✓
A7 – Broken Authentication & Session Management				
A8 – Insecure Cryptographic Storage				
A9 – Insecure Communications				
A10 – Failure to Restrict URL Access		✓ +	✓ *	

* Spring Security + HDIV

** Spring Web Flow

*** ORM (Hibernate)

Quelle: http://www.owasp.org/index.php/Top_10_2007

Jasypt – Java simplified encryption

- Implementiert die aktuellen Sicherheitsstandards
 - Digest-Erstellung
 - Texte, Zahlen oder Binär-Daten mit Hilfe eines Passwortes verschlüsseln
- Leichte Integration in:
 - Hibernate 3
 - Spring
 - Spring Security
 - Bouncy Castle

Jasypt – Spring-Integration

```
<bean id="strongEncryptor",  
      class="org.jasypt.encryption.pbe.StandardPBEStrategyEncryptor">  
  <property name="algorithm">  
    <value>PBEWithMD5AndTripleDES</value>  
  </property>  
  <property name="password">  
    <value>jasypt</value>  
  </property>  
</bean>
```

```
class MyEncrypter{  
  private StandardPBEStrategyEncryptor enc = null;  
  ...  
  public void setEnc(StandardPBEStrategyEncryptor nEnc){this.enc = nEnc;}  
  ...  
  public String doEncrypt(String value){  
    return enc.encrypt(value);  
  }  
  ...  
}
```

Jasypt – ORM-Integration (Hibernate 3)

```
<hibernate-mapping package="myapp">
  ...
  <typedef name="encryptedString"
    class="org.jasypt.hibernate.type.EncryptedStringType">
    <param name="algorithm">PBEWithMD5AndTripleDES</param>
    <param name="password">jasypt</param>
    <param name="keyObtentionIterations">1000</param>
  </typedef>
  ...
  <class name="UserData" table="USER_DATA">
    ...
    <property name="address" column="ADDRESS" type="encryptedString" />
    ...
  </class>
  ...
</hibernate-mapping>
```

➤ Auch mittels Annotationen

OWASP Top Ten (2007)

	Jasypt	Erw. 2	Erw. 1	MVC
A1 – Cross Site Scripting	Jasypt	✓ +		✓
A2 – Injection Flaws		✓ +	✓ ***	✓
A3 – Malicious File Execution				
A4 – Insecure Direct Object Reference		✓ +	✓ *	
A5 – Cross Site Request Forgery		✓ +	✓ **	
A6 – Information Leakage & Improper Error Handling				✓
A7 – Broken Authentication & Session Management				
A8 – Insecure Cryptographic Storage		✓ ++		
A9 – Insecure Communications				
A10 – Failure to Restrict URL Access		✓ +	✓ *	

- * Spring Security + HDIV
- ** Spring Web Flow ++ Jasypt
- *** ORM (Hibernate)

Quelle: http://www.owasp.org/index.php/Top_10_2007

MVC & OWASP Top Ten (2007)

	Erw. 2	Erw. 1	MVC
A1 – Cross Site Scripting	✓ +		✓
A2 – Injection Flaws	✓ +	✓ ***	✓
A3 – Malicious File Execution			
A4 – Insecure Direct Object Reference	✓ +	✓ *	
A5 – Cross Site Request Forgery	✓ +	✓ **	
A6 – Information Leakage & Improper Error Handling			✓
A7 – Broken Authentication & Session Management			
A8 – Insecure Cryptographic Storage	✓ ++		
A9 – Insecure Communications			
A10 – Failure to Restrict URL Access	✓ +	✓ *	

- * Spring Security + HDIV
- ** Spring Web Flow ++ Jaspyt
- *** ORM (Hibernate)

Quelle: http://www.owasp.org/index.php/Top_10_2007

Literatur

- OWASP: www.owasp.org
- Spring: www.springframework.org
- Spring Security: www.acegisecurity.org
- Spring Web Flow: www.springsource.org/webflow
- HDIV: www.hdiv.org
- Jasypt: www.jasypt.org

Vielen Dank für die Aufmerksamkeit!

? Fragen ?