OWASP Stammtisch München

Sep 2014

XSS und andere Sicherheitslücken aus der Perspektive des Programmcodes

RIGS IT

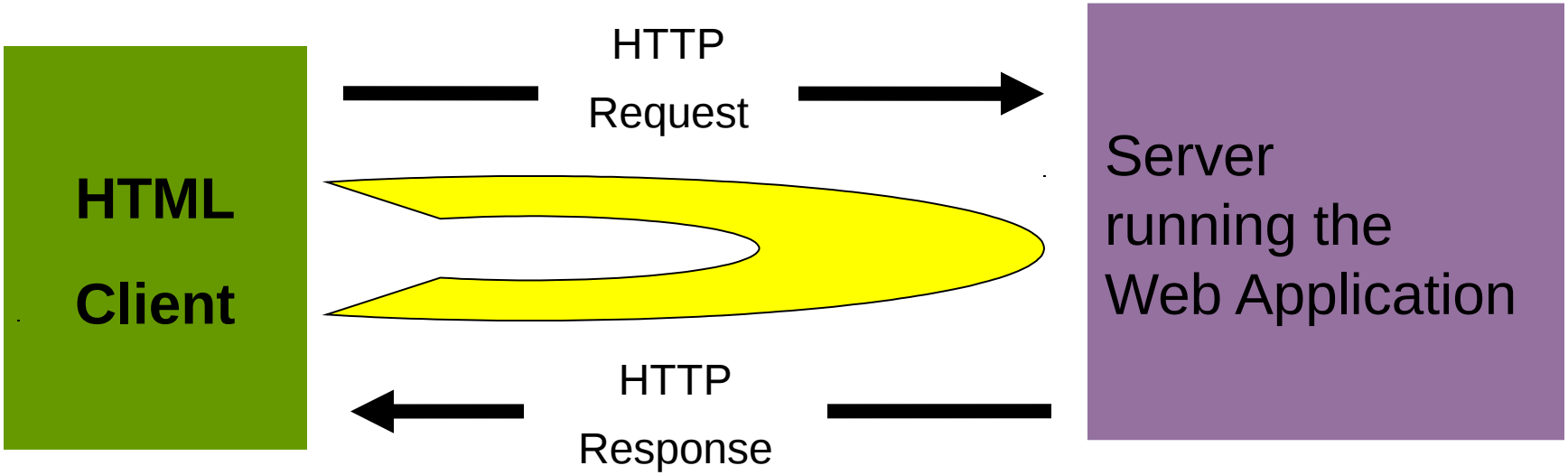
XSS: Cross-Site Scripting

1.) Es gelangen Daten in den Web-Browser, die Steuerungsinformationen enthalten (HTML, JavaScript, VBScript, ActiveX, Flash, ...) und vom Browser ausgeführt werden.

2.) Wobei die Datenquelle von einem Angreifer kontrolliert wird.

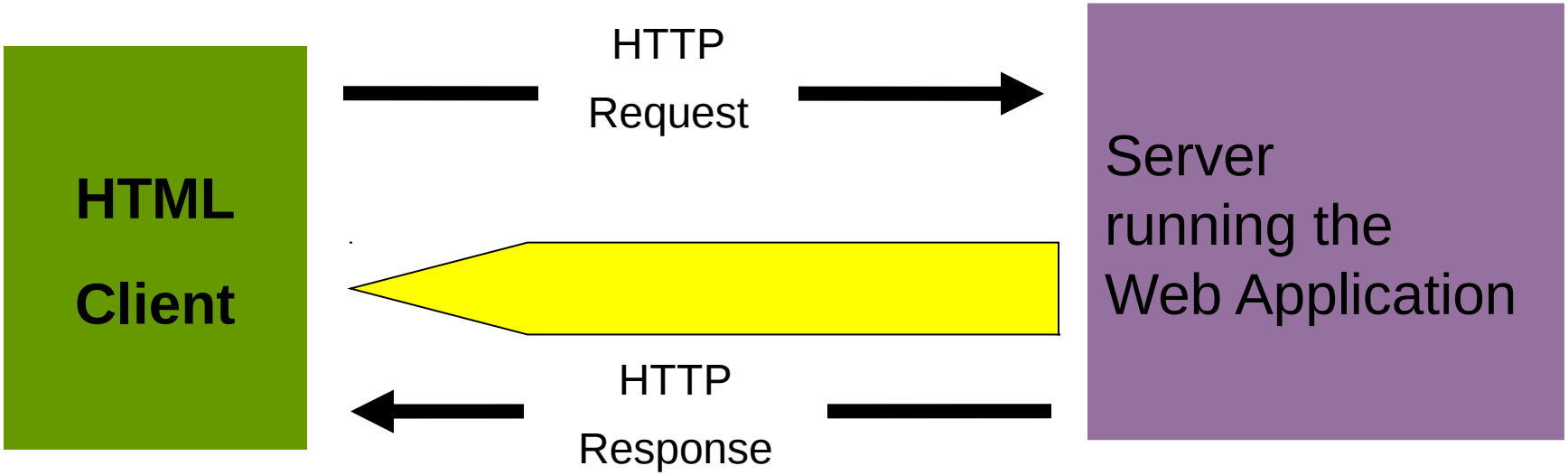
3.) 3 Varianten

Reflected XSS



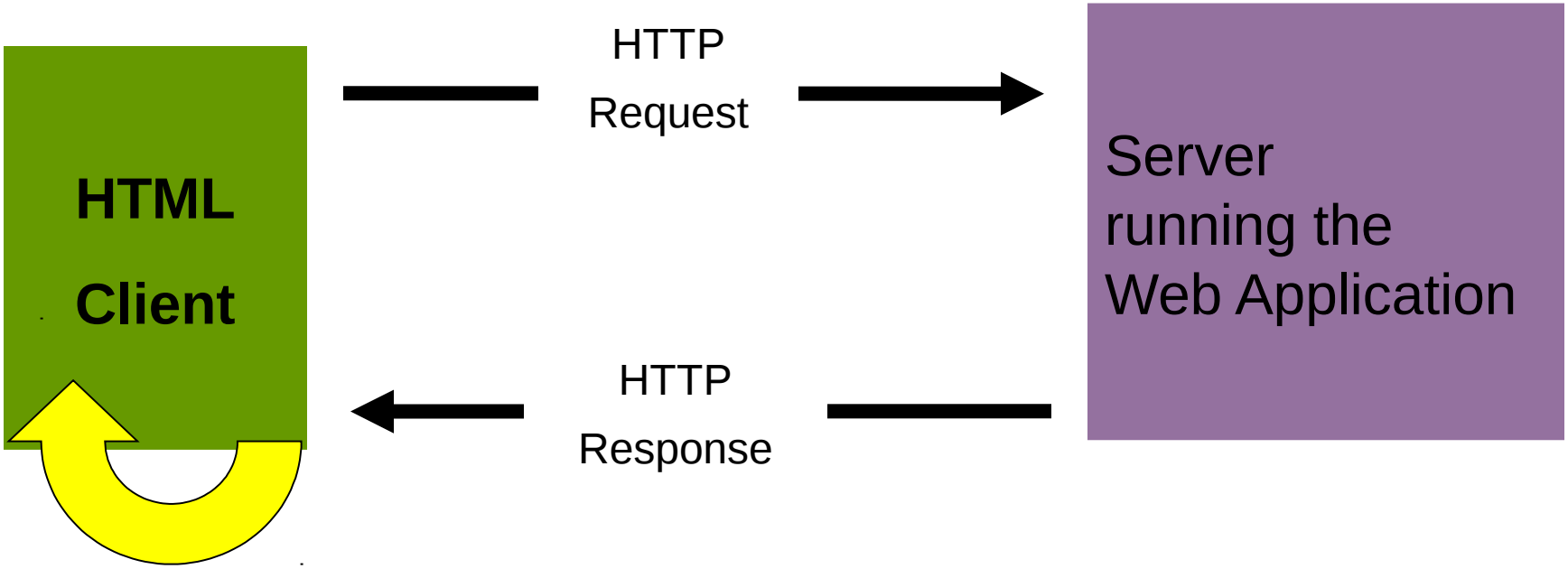
Der Schadcode wird vom Browser mit dem Request an den Webserver gesendet und kommt von diesem mit dem Response zurück an den Browser (reflektiert).

Stored XSS / Persistent XSS



Der Schadcode ist schon auf dem Webserver gespeichert und kommt von diesem mit dem Response zu dem Browser.
Beispiel: Gästebuch, Forum

DOM-based XSS / Local XSS



Der Schadcode kommt nicht vom Webserver. Er gelangt erst nach dem Laden der Webseite in den Browser durch Script-code, der den DOM (Document Object Model „Tree“) ändert.

z.B. durch einen angehängten Parameter oder URI-Fragment in der URL

XSS Example mit JSP

Weitere Security Schwachstellen am Beispiel der OWASP Webgoat Applikation

Reales Projekt:

Open source Java eCommerce platform based on the Spring Framework

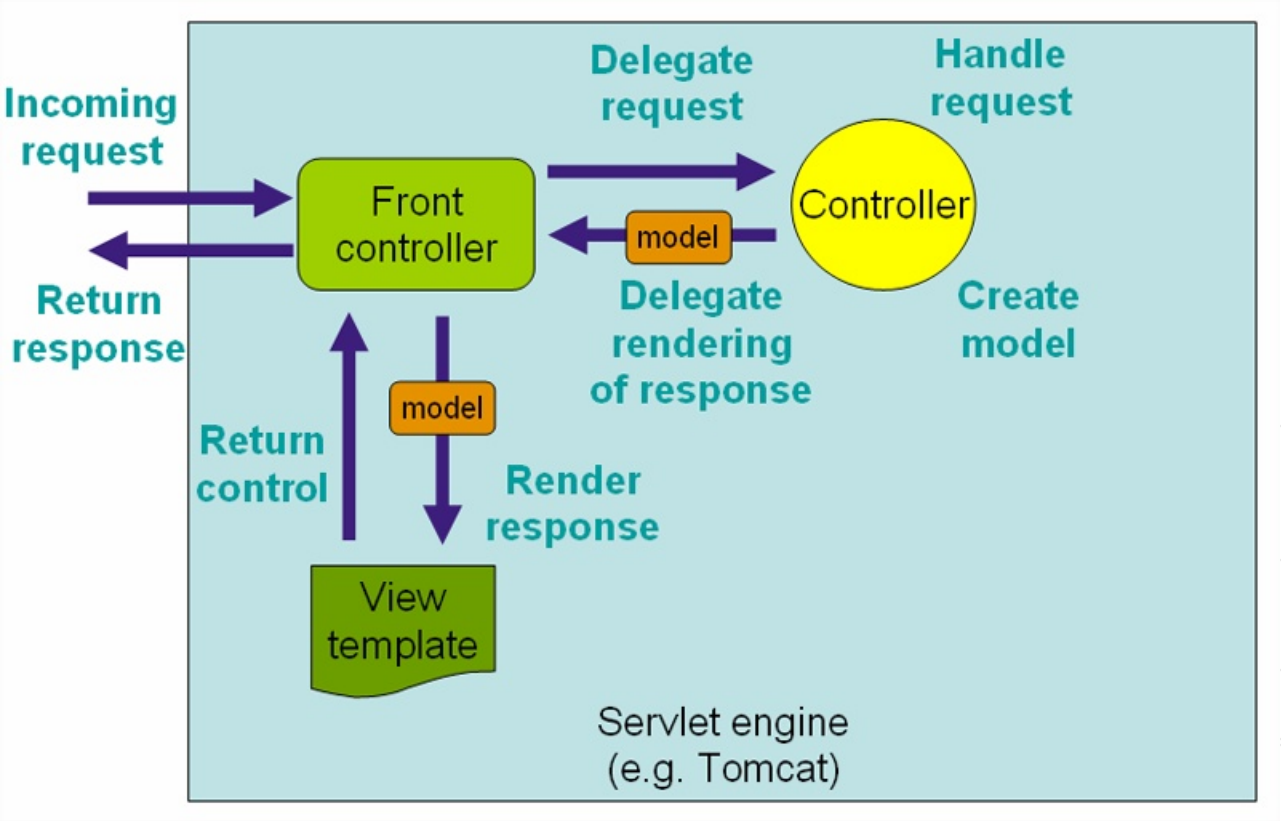
broadleaf

XSS: Beispiel in petclinicplus

(Demo Applikation für Spring und freemarker)

Beispiel: Request/Response-Zyklus

- Spring IOC Framework
- Spring Web MVC Framework
- Freemarker "Template Engine"



Quelle: Dokumentation von Spring V 2.5

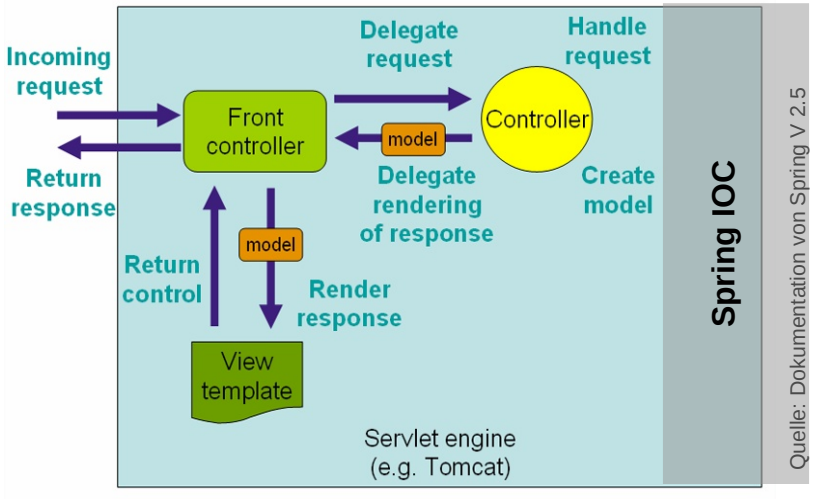
Spring IOC Framework

Inversion of Control (IoC): Das Programm ruft nicht die Library auf, sondern die Library ruft das Programm auf.

Ausgewählte Objekte (Beans) des Programms werden nicht vom Programm selbst sondern von Spring instantiiert und mit Eigenschaften (Properties) versehen.

Spring-Beans werden in einem IoC-Container verwaltet und bei Bedarf an das Programm übergeben (injected).

Properties werden über XML Dateien oder Java-Annotations konfiguriert



Spring MVC Web Framework (Ver 2)

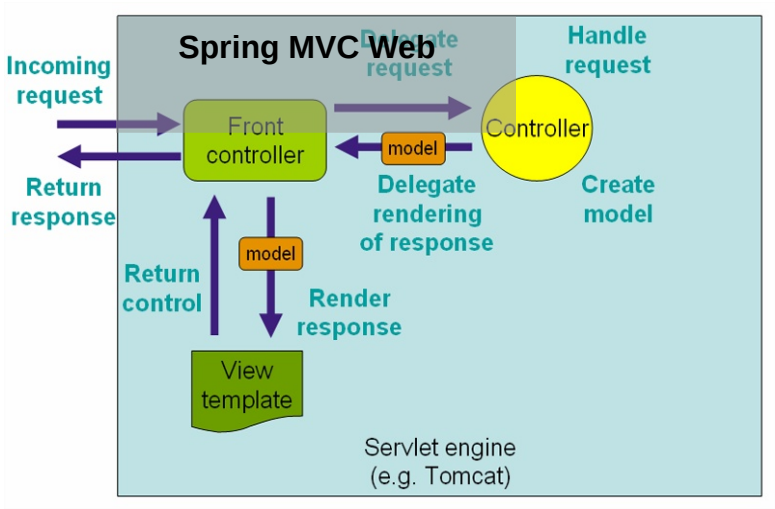
Implementiert das Model-View-Controller Pattern für Web-Applikationen

Requests werden dispatched zu Controller-Klassen, die den Request verarbeiten

Stützt sich auf Spring IOC-Container

Ver 2: Controller liefern ein ModelAndView Objekt zurück

```
<!--
   This bean is an explicit URL mapper that is used by the "petclinicplus" DispatcherServlet
   It is used instead of the default BeanNameUrlHandlerMapping.
-->
<bean id="urlMapping"
   class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
   <property name="mappings">
      <props>
         <prop key="/welcome.htm">clinicController</prop>
         <prop key="/vets.htm">clinicController</prop>
         <prop key="/findOwners.htm">findOwnersForm</prop>
         <prop key="/owner.htm">clinicController</prop>
         <prop key="/addOwner.htm">addOwnerForm</prop>
         <prop key="/editOwner.htm">editOwnerForm</prop>
         <prop key="/addPet.htm">addPetForm</prop>
         <prop key="/editPet.htm">editPetForm</prop>
         <prop key="/addVisit.htm">addVisitForm</prop>
         <prop key="/logout.htm">authController</prop>
         <prop key="/login.htm">authController</prop>
         <prop key="/loginerror.htm">authController</prop>
      </props>
   </property>
</bean>
```



Quelle: Dokumentation von Spring V 2.5

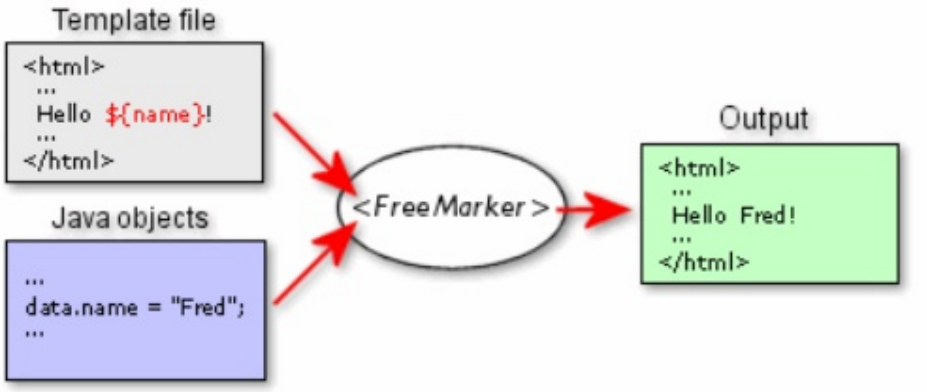
Freemarker

Freemarker ist eine Template Engine, die als Komponente innerhalb eines Web Applikationsframeworks HTML Dateien aus Templates für den Response Output erzeugt.

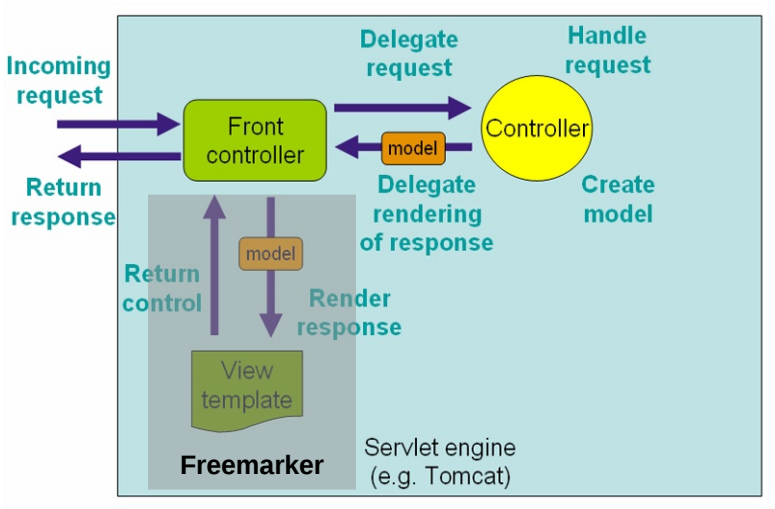
FTL-Templates werden mit Daten aus Java Objekten befüllt.

Eingeschränkte Programmierung möglich

Open Source



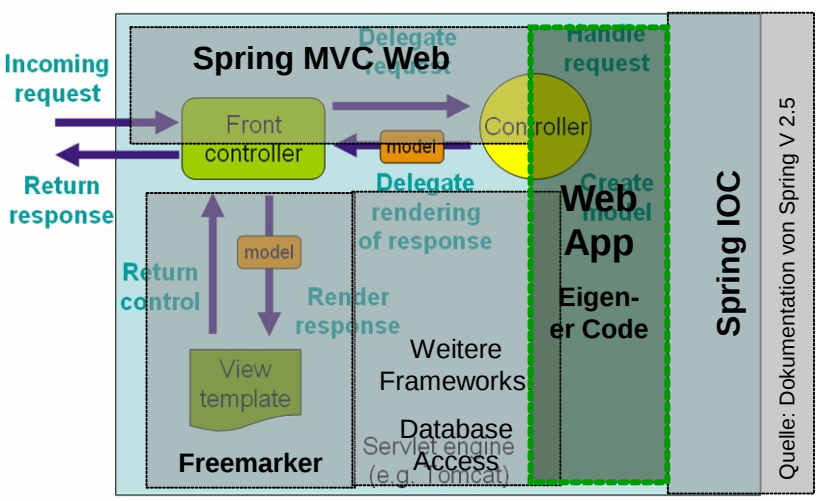
Quelle: Dokumentation von Freemarker



Quelle: Dokumentation von Spring V 2.5

Analyse einer Web Applikation

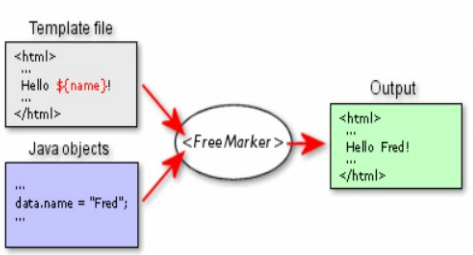
Grossteil der Applikation besteht aus Frameworks



Simulation für statische Analyse:

- Spring IOC: Instantiierung der Spring Beans mit Aufruf von Konstruktoren und Settern
- Spring Web: Instantiierung von Controller, Command, Interceptor, Validator Klassen; Aufruf von custom Methoden in diesen Klassen
- Freemarker: Zugriffe auf Java-Datenobjekte in FTL-Templates mit Berücksichtigung auf Encoding; Mapping von Controller-Klasse zu FTL-Template

Freemarker Encoding



Quelle: Dokumentation von Freemarker

HTML Encoding (Sanitizing) im Template möglich:

- BuiltIn Operatoren: ?html, ?js_string, ?url
- Blockweise Direktiven: <#escape> und <#noescape>
- liefern Daten (potentiell tainted) an die eigene Webapplikation

```
<TABLE border="0">
  <TR><TD>Name</TD><TD><b>${owner.firstName} ${owner.lastName}</b></TD></TR>
  <TR><TD>Address</TD><TD>${owner.address?html}</TD></TR>
  <TR><TD>City</TD><TD>${owner.city}</TD></tr>
  <TR><TD>Telephone</TD><TD>${owner.telephone}</TD></TR>
  <TR>
```