JEE Security

Misc Slides #2

JEE Security

Security is another vast topic, we just have a short look at **authentication** (who are you?) and **authorization** (what can you do?)

Application managed
- Do it yourself, ... by the way are you a security pro??

Container managed
- Security implemented by pros!

Also many low level API's, we don't

Some JEE Security API's

Java Authentication and Authorization Service (JAAS)
Java Cryptography Extension (JCE)
Java Generic Security Services (Java GSS-API)
Java Secure Socket Extension (JSSE)
Simple Authentication and Security Layer (SASL)
XML Digital Signature

........much more, phuiii...

But we dont'...

JEE Application Managed Security

Do it yourself (as in Workshop 1, ok for this course)

- Put "everything" in private parts of application (below WEB-INF directory)
- Only visible pages: home/login/logout
- Use a filter (javax.servlet.Filter) to protect resources (URI's)
    - Filter can forward, redirect
- Users and groups (roles) in database tables

JEE Container Managed Security

Predefined JEE security design/system handled by containers. Security configuration Server dependent

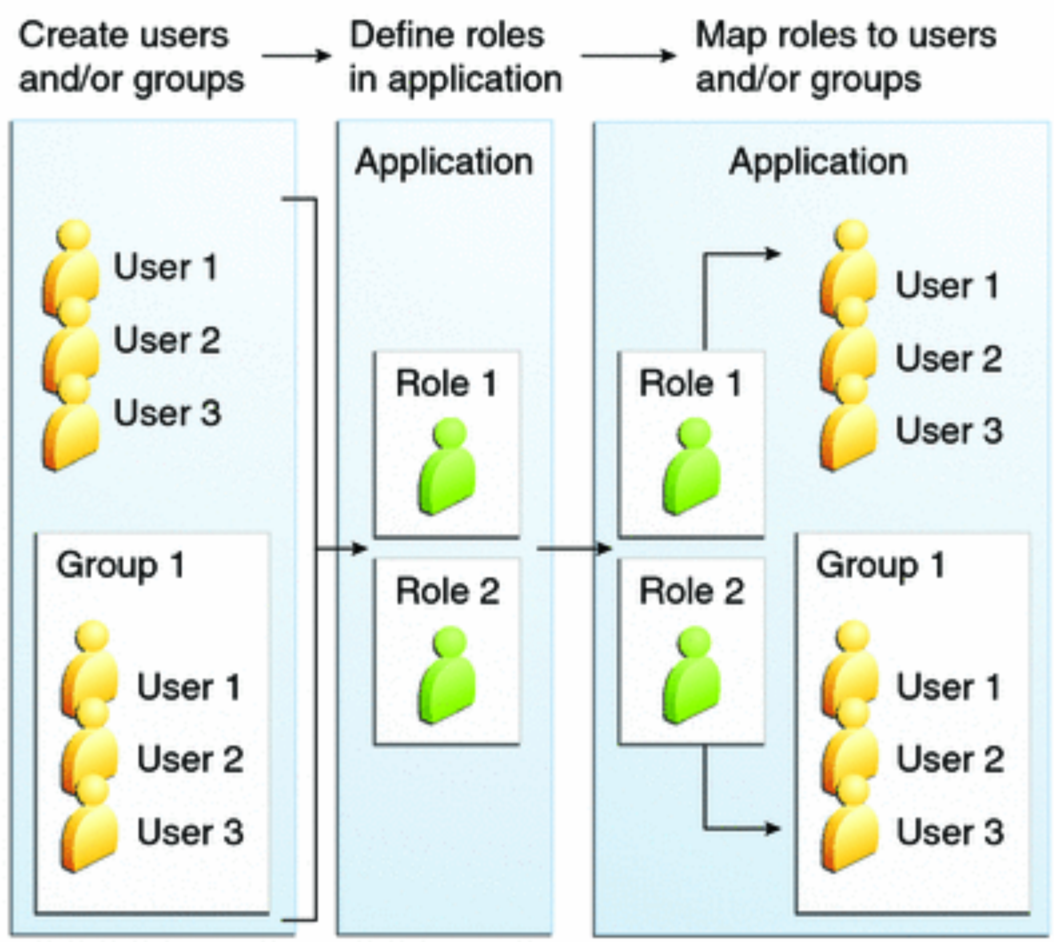
Possibilities

- Securing Web Application
- Securing EJB's, … **not covered**

Realms, Users, Groups, and Roles

A realm is a security policy domain defined for a web or application server. A realm contains a collection of users, who may or may not be assigned to a group (implemented as file-, jdbc-, LDAP-realms, ...)

A role is an abstract name for the permission to access a particular set of resources in an application.



Web Security Constraints

Web resource collection: A list of URL patterns (the part of a URL after the hostname and port you want to constrain) and HTTP operations (the methods within the files that match the URL pattern you want to constrain) that describe a set of resources to be protected.

Authorization constraint: Specifies whether authentication is to be used and names the roles authorized to perform the constrained requests.

User data constraint: Specifies how data is protected when transported between a client and a server.

Specified in web.xml

Web Security Constraints Example

// web.xml

```
<security-constraint>
    <web-resource-collection>
        <web-resource-name>wholesale</web-resource-name>
        <url-pattern>/acme/wholesale/*</url-pattern>
        <http-method>GET</http-method>
        <http-method>POST</http-method>
    </web-resource-collection>
    <auth-constraint>
        <role-name>PARTNER</role-name>  <!-- Role name in application -->
    </auth-constraint>
    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>
```

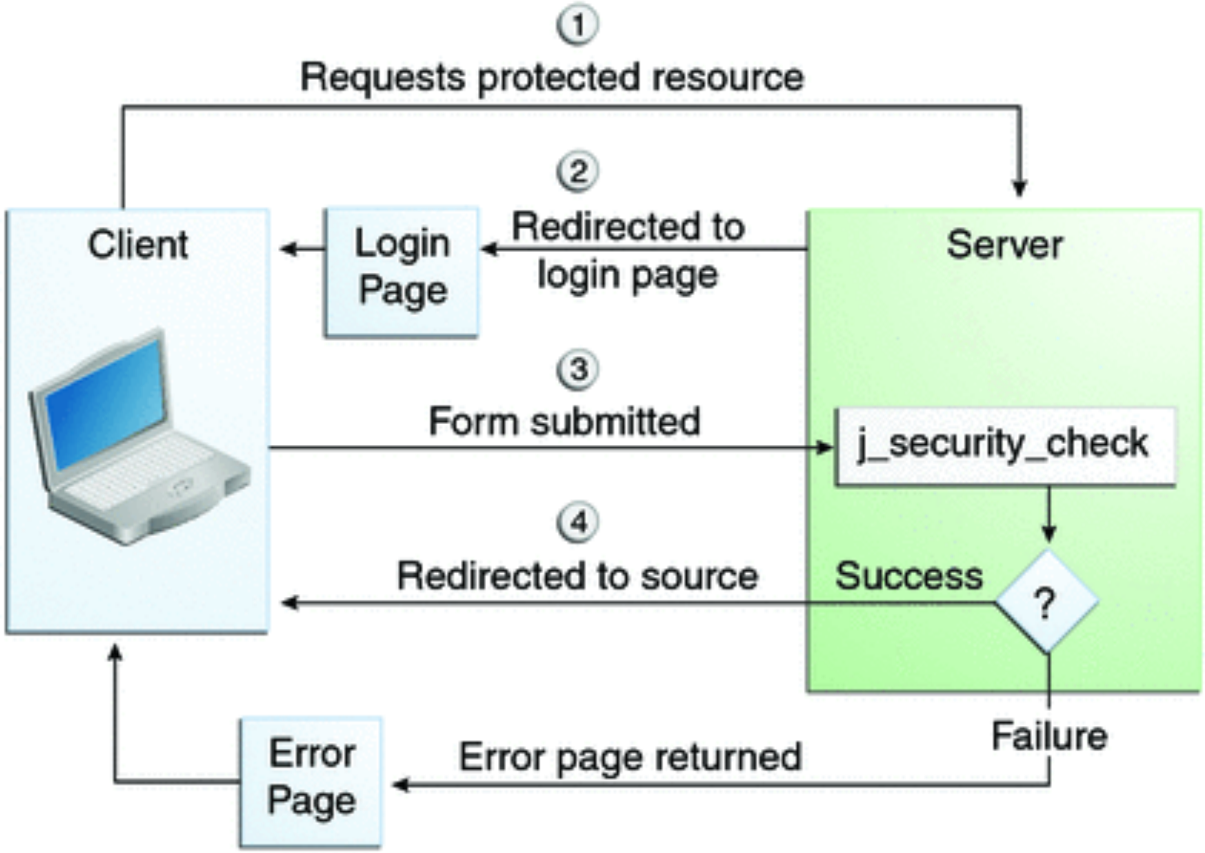
CONFIDENTIAL = GlassFish will use SSL (alt. NONE)

Web Authorization Mechanism

JEE supports

-Basic authentication
-Form-based authentication, the **only covered in course**
-Digest authentication
-Client authentication
-Mutual authentication

Form Based Authorization



Specify Authorization Mechanism

```
// web.xml
<login-config>
    <auth-method>FORM</auth-method>
    <realm-name>file</realm-name>
    <form-login-config>
        <form-login-page>/login.xhtml</form-login-page>
        <form-error-page>/error.xhtml</form-error-page>
    </form-login-config>
</login-config>

 <error-page>
        <!-- Access denied (bad role response) -->
        <error-code>403</error-code>
        <location>/notAuthorized.xhtml</location>
 </error-page>
```

Setup with Database and GlassFish

Must have database (some column must be specified as login and password columns and more...)

Must have a DataSource (in file Other Sources/setup/glassfish-resources.xml, ... wizard in NetBeans)

Create the GlassFish (JDBC) realm
- Use Admin Console

Must map roles in application to roles in GlassFish, in WEB-INF/glassfish-web.xml (many application can use same server roles)

See code sample