

# Java EE Security

# Zabezpečení aplikace

- Při zabezpečení Vaší webové aplikace máte několik možností:
  - Programově pomocí vlastního security filtru.
  - Java EE security.
  - Spring security.
  - Security pomocí dalších frameworků.
- V této přednášce se zaměříme na Java EE security.
- Ať se rozhodnete pro jakýkoli typ security, tak budete řešit čtyři body bezpečnosti:

- **Authentication** – je klient skutečně tím, co říká?

→ Zadal správnou kombinaci jméno / heslo?

- **Authorization** – má klient příslušná práva?

→ Uživatel s rolí USER  
By neměl přistupovat  
Do administrátorské sekce

- **Confidentiality** – není možné číst obsah dat

- **Data integrity** – není možné podvrhnout data

→ Řeší SSL  
(HTTPS protokol)

# Uživatelská jména, hesla, role

- Někde musíme mít definované uživatele s heslem a rolemi, které se k jejich uživatelskému jménu pojí.
- V Java EE security máme několik možností:
  - Soubor `[apache-tomcat]/conf/tomcat-users.xml` – jednoduchý soubor, kde jsou pomocí XML definované role a uživatelé, tento způsob je vhodný v raných fázích programování aplikace.
  - Databáze
  - LDAP
  - A další ...
- Soubor `tomcat-users.xml` je výchozí zdroj uživatelských jmen a rolí. Způsob získávání těchto dat je možné změnit v `[apache-tomcat]/conf/server.xml` – hledejte `<Realm>`, viz.  
<http://tomcat.apache.org/tomcat-6.0-doc/realms-howto.html>

# Soubor tomcat-users.xml

- Níže je ukázkový obsah souboru tomcat-users.xml:

```
<tomcat-users>  
  <role rolename="tomcat"/>  
  <role rolename="user"/>  
  <role rolename="admin"/>  
  
  <user username="tomcat" password="tomcat" roles="tomcat"/>  
  <user username="jirka" password="pinkas" roles="user"/>  
  <user username="admin" password="admin" roles="tomcat,admin"/>  
</tomcat-users>
```

Definice rolí

Definice uživatelů

- Nevýhoda souboru tomcat-users.xml: Jeho obsah se načítá jednou při startu Tomcatu, tudíž při jeho změně je nutné restartovat Tomcat!

# Způsob přihlášení

- Následně musíme v naší aplikaci ve web.xml souboru definovat jaký způsob přihlašování budeme používat:

```
<login-config>
```

```
    <auth-method>BASIC</auth-method>
```

```
</login-config>
```

Absence šifrování se  
řeší pomocí HTTPS

- Existují čtyři typy přihlášení:
  - BASIC:** přenáší data nezašifrovaně
  - FORM:** podobné jako BASIC, ale umožňuje zadat vstupní data (jméno/heslo) do formuláře
  - DIGEST:** přenáší data šifrovaně, ale není moc používané
  - CLIENT-CERT:** autentizace pomocí klientského certifikátu. Jedná se o velice bezpečný způsob autentizace, ale klient musí mít před použitím na svém počítači certifikát

# Definování rolí ve web.xml

- Webová aplikace musí mít někde informaci jaké role se v ní budou používat (tyto role odpovídají našim rolím definovaným v souboru tomcat-users.xml, databázi, LDAPu atd.).
- Role používané v aplikaci se definují ve web.xml:

```
<security-role>  
    <role-name>user</role-name>  
</security-role>  
  
<security-role>  
    <role-name>admin</role-name>  
</security-role>
```

# Definování omezení u resources

- Nyní již můžeme omezit přístup k jednotlivým URL v aplikaci:

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>admin zone</web-resource-name>
    <url-pattern>/admin/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>admin</role-name>
  </auth-constraint>
</security-constraint>
```

# Form přihlášení

## ① In the DD...

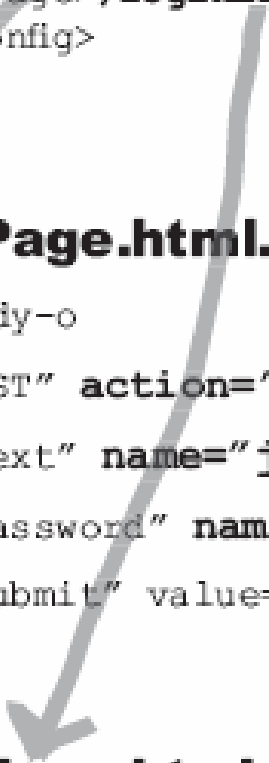
```
<login-config>
  <auth-method>FORM</auth-method>
  <form-login-config>
    <form-login-page>/loginPage.html</form-login-page>
    <form-error-page>/loginError.html</form-error-page>
  </form-login-config>
</login-config>
```



## ② Inside the loginPage.html...

```
Please login daddy-o

<form method="POST" action="j_security_check">
  <input type="text" name="j_username">
  <input type="password" name="j_password">
  <input type="submit" value="Enter">
</form>
```



## ③ Inside the loginError.html...

```
<html><body>
  Sorry dude, wrong password
</body></html>
```



# Vynucení HTTPS

- V Tomcatu je nutné nastavit <Connector> pro port 8443 s podporou SSL.
- Do tagu <security-constraint> vložte následující kód:

```
<user-data-constraint>
```

```
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
```

```
</user-data-constraint>
```

- V <transport-guarantee> mohou být následující hodnoty:
  - **NONE** – default (žádná bezpečnost dat)
  - **INTEGRAL** – data se nesmí po cestě změnit
  - **CONFIDENTIAL** – data nesmí být nikým čtena

Je jedno, co  
uvedete, obojí  
bude mít stejný  
výsledek