Java technológiák - 9. előadás.

Webalkalmazások biztonsága.

ANTAL Margit

Sapientia - EMTE

2010

Témakörök

- Biztonsági mechanizmusok
- Alaphitelesítés és autorizáció
- Űrlap alapú hitelesítés
- HTTPS
- Programozott biztonsági mechanizmusok

Alapfogalmak

- Authentication - Hitelesítés
- Authorization - Jogosultság vizsgálat
- Confidentiality - Bizalmasság
- Integrity - Adatintegritás

Authentication - Hitelesítés

- Kérdés: Ki vagy te?
- Válasz:
  - felhasználónév és jelszó
  - biometriás azonosságvizsgálat:
    - ujjlenyomat,
    - arc,
    - DNS,
    - tenyér,
    - retina,
    - hang,
    - billentyűzési ritmus,
    - testmozgás

Authorization - Jogosultság vizsgálat

- Kérdés: Milyen erőforrásokhoz férhet hozzá a felhasználó?
- Tulajdonságok
  - Csak hitelesített felhasználókra
  - A jogosultságok szerepkörökre bonthatók
  - A jogosultságok deklaratív módon is megadhatók (telepítésleíróban)

Integrity - Adatintegritás

- Az adatok küldés közben nem változnak meg
- Megoldás: az adatokhoz hozzátesznek egy ellenőrző összeget.

Confidentiality -Bizalmasság

- Az adatok védelme az arra nem jogosultaktól
- Ha szállítás közben idegen kézbe kerülnek az adatok, azok jelentését ne lehessen megfejteni
- Megoldás: Titkosítás

Adatintegritás + Bizalmasság

- A gyakorlatban egy szállításszintű protokollt használnak.
- SSL = Secure Sockets Layer
- Java EE szinten ez az alkalmazásszerver konfigurálásához tartozik

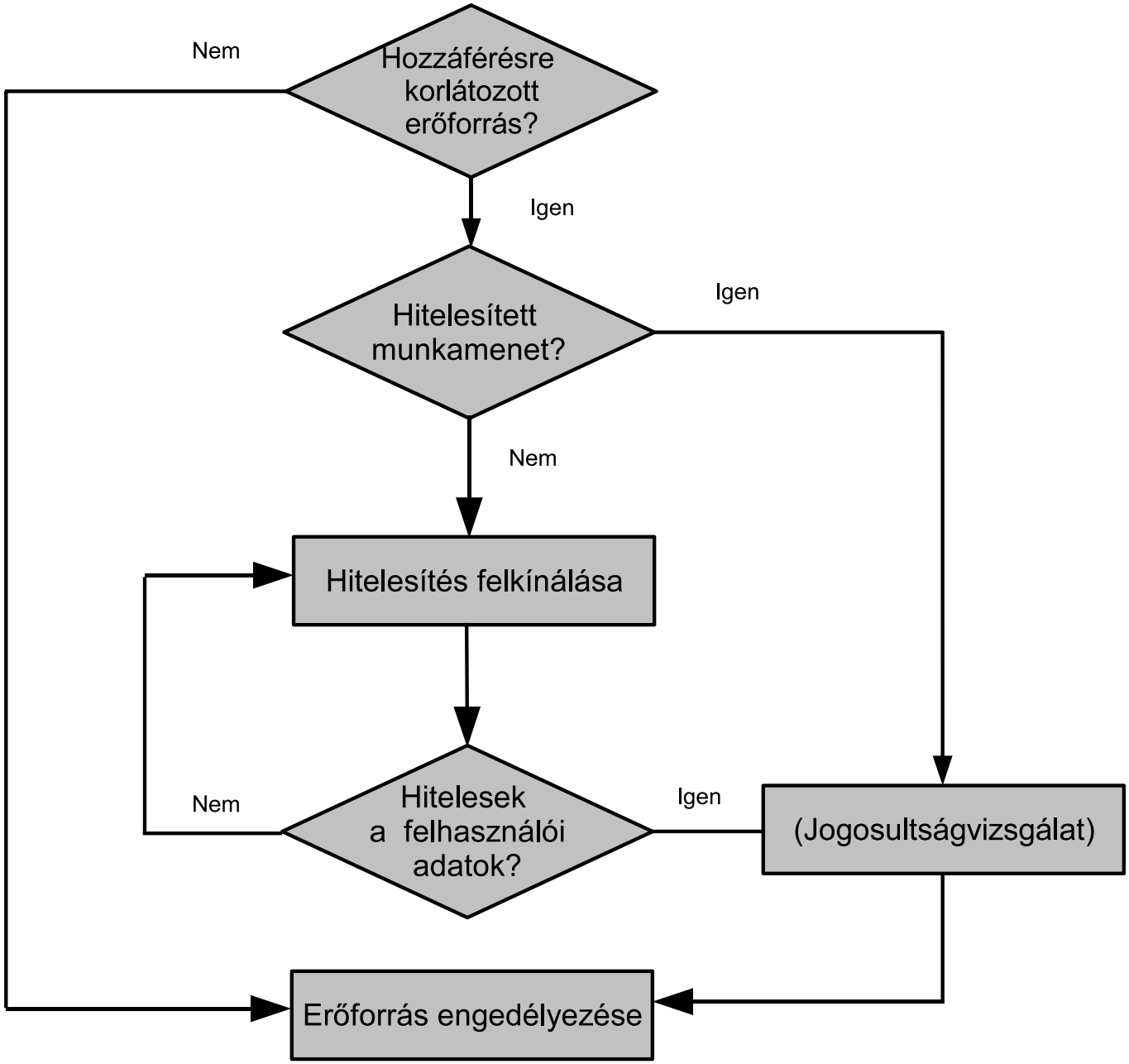
Hogyan szabályozhatjuk?

- Deklaratív módszer
- Programozott módszer

Deklaratív módszer

- Az alkalmazás telepítésleírójában szabályozzuk
- A webkonténer végrehajtja
  - Tipikusan Lusta hitelesítést végez
  - A lusta hitelesítés (lazy authentication) csak akkor kéri a felhasználót hitelesítésre, ha védett erőforrást akar elérni

Hitelesítés a Webrétegben



A WEB réteg hitelesítési mechanizmusai

- HTTP alaphitelesítés - BASIC
- HTTP digest hitelesítés - DIGEST
- Űrlap alapú - FORM
- HTTPS ügyfélhitelesítés - CLIENT CERT

A Java EE specifikáció három hitelesítési mechanizmust ír elő a webréteg számára:

- alaphitelesítés
- űrlap alapú hitelesítés
- ügyfélhitelesítés

Java EE absztrakt szerepkörök

- A Java EE specifikálja, hogy az alkalmazásszervereknek támogatniuk kell a szerepek alapján történő jogosultságvizsgálatot.
- Az absztrat szerepkör egy karakterlánc; például: admin, user,...
- Az alkalmazás fejlesztője csak absztrakt szerepkörökkel dolgozik
- Telepítéskor megadjuk az absztrakt szerepkörök leképzését konkrét felhasználókra és csoportokra

realm - user registry

A felhasználók hitelesítési adatait és csoporttagságait nyilvántartó adattároló
Példa realm-re:

- Sun Glassfish: file, certificate
- IBM WebSphere: LDAP szerver vagy lokális op. rendszer
- Saját realm is fejleszthető

HTTP alaphitelesítés - BASIC

- A böngészőt utasítja az alkalmazás a hitelesítési adatokat tartalmazó ablak megjelenítésére.
- A felhasználónév-jelszó párost a böngésző eljuttatja a szerverhez
- Ez a legegyszerűbb hitelesítési mechanizmus
- A felhasználónév-jelszó páros beviteli ablak formáját a böngésző határozza meg

web.xml

```
<login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>file</realm-name>
</login-config>
```

A BASIC hitelesítés lépései

1. A Glassfish alkalmazásszerverben hozzuk létre a fizikai felhasználókat és csoportokat.
2. Az alkalmazás telepítésleírójában (web.xml):
   - A hitelesítési mechanizmus kiválasztása.
   - Az absztrakt szerepkörök deklarálása.
   - Védett előforrások deklarálása.
3. Az alkalmazás specifikus telepítésleírójában (sun-web.xml vagy glassfish-web.xml) az alkalmazásban szereplő absztrakt szerepköröket leképezzük az alkalmazásszerverben (Glassfish) létrehozott felhasználókra, illetve felhasználócsoportokra.

1. Glassfish alkalmazászerverben

Hozzuk létre a következő felhasználókat:

- Username: admin1; Group: administrators
- Username: admin2; Group: administrators
- Username: admin3; Group: administrators

2. web.xml

- 2.1 Hitelesítési mechanizmus meghatározása
- 2.2 Absztrakt szerepkörök meghatározása
- 2.3 Védett erőforrások meghatározása

2.1 Hitelesítési mechanizmus meghatározása

```
<login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>file</realm-name>
</login-config>
```

2.2 Absztrakt szerepkörök meghatározása

```
<security-role>
    <role-name>adminisztratorok</role-name>
</security-role>
<security-role>
    <role-name>felhasznalok</role-name>
</security-role>
```

2.3 Védett erőforrások meghatározása

```
<security-constraint>
    <display-name>Constraint1</display-name>
    <web-resource-collection>
        <web-resource-name>
            JSPlapok
        </web-resource-name>
        <description/>
        <url-pattern>*.jsp</url-pattern>
        <http-method>GET</http-method>
        <http-method>POST</http-method>
    </web-resource-collection>
    <auth-constraint>
        <description/>
        <role-name>adminisztratorok</role-name>
    </auth-constraint>
</security-constraint>
```

3. sun-web.xml (glassfish-web.xml)

Absztrakt szerepkörök leképzése konkrét felhasználókra:

- role-name: absztrakt szerepkör (web.xml)
- group-name: fizikai csoportnév (Glassfish alkalmazásszerver)
- principal-name: a Glassfish alkalmazásszerverben meghatározott fizikai csoportnév

```
<security-role-mapping>
        <role-name>adminisztratorok</role-name>
        <group-name>administrators</group-name>
</security-role-mapping>
```

vagy

```
<security-role-mapping>
        <role-name>adminisztratorok</role-name>
        <principal-name>admin1</principal-name>
        <group-name>administrators</group-name>
</security-role-mapping>
```

A BASIC hitelesítés hátrányai

- A hitelesítési adatok bekérése böngészőspecifikus. (nem illeszkedi a weblaphoz!)
- A kijelentkezés nehézkes (böngészőspecifikus).
- A hitelesítési adatok küldése Base64 kódolással történik.

Űrlap alapú hitelesítés

- A fejlesztő határozza meg a hitelesítési űrlap formáját (a hibalapot is!).
- Ha védett erőforrásra irányul a kérés, a webkonténer előbb a hitelesítő űrlapot küldi.
- A hitelesítést a konténer végzi.
- Ugyanolyan biztonságos, mint az alaphitelesítés.

Űrlap alapú hitelesítés – web.xml

Kötelező megadni a következőket:

- egy bejelentkezési űrlap: formlogin.jsp
- egy hibalap: formloginerror.jsp

```
<login-config>
  <auth-method>FORM</auth-method>
  <realm-name>file</realm-name>
  <form-login-config>
    <form-login-page>
       /formlogin.jsp
    </form-login-page>
    <form-error-page>
       /formloginerror.jsp
    </form-error-page>
  </form-login-config>
</login-config>
```

Űrlap alapú hitelesítés - beállítások

Minden beállítás ugyanúgy történik, mint a BASIC hitelesítésben, kivéve a következőket:

- El kell készíteni a bejelentkezési űrlapot (formlogin.jsp).

```
<form action="j_security_check" method="post">
  Nev   : <input type="text"
                 name="j_username"><br><br>
  Jelszo: <input type="password"
                 name="j_password"><br><br>
  <input type="submit" value="Elkuld"/>
</form>
```

- El kell készíteni a hiba űrlapot

```
<h1>Hiba a bejelentkezesnel!</h1>
```

- Az alkalmazás telepítésleírójában a FORM típusú hitelesítést kell megadni

Magas szintű biztonság: HTTPS

```
<security-constraint>
  <web-resource-collection>
    ...
  </web-resource-collection>

  <auth-constraint>
    ...
  </auth-constraint>

  <user-data-constraint>
    <transport-guarantee>
       CONFIDENTIAL
    </transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

A $<$ *transport − guarantee*/ $>$ tag

A $<$ *transport − guarantee* $>$ tag a következő értékeket veheti fel:

- NONE
- INTEGRAL
- CONFIDENTIAL

Megjegyzések:

- Az INTEGRAL vagy a CONFIDENTIAL beállítás bármelyike garantálja a HTTP over SSL, vagyis a HTTPS protokoll használatát.
- Használható mindkét hitelesítési típussal: BASIC, FORM

Glassfish beállítások HTTPS használatához

Glassfish 3.1

Configurations → server-config → Network Config → Network Listeners → http-listener-1 → SSL fül

Deklaratív vagy programozott biztonság

- Deklaratív biztonság:
  - egyszerű
  - hibalehetőség minimális
  - nem igényel programozói ismereteket (csak XML)
- Programozott biztonság:
  - ha a biztonsági beállítások időben dinamikusan változnak (pl. játék különböző szintjein különböző erőforrásokhoz van hozzáférés)
  - programozói ismereteket igényel

Deklaratív vagy programozott biztonság

- a programozott biztonság együtt tud működni a deklaratív biztonsággal, sajátosíthatja azt
- HttpServletRequest interfész:
  - boolean isUserInRole(String absztrakt-szerepkor)
  - String getRemoteUser() –null, ha még nem történt meg a hitelesítés

Tanusítvány alapú biztonság

Miért van szükség?

- Nemcsak személy típusú felhasználó létezik, hanem program is igénybe veheti egy másik program szolgáltatásait
- A tanusítvány nemcsak felhasználót és jelszót tartalmazhat

Tanusítvány ellenőrzése

- Kliens hitelesítés: a szerver ellenőrzi az ügyfél hitelességét
- Szerver hitelesítés: a kliens ellenőrzi a szerver hitelességét
- kölcsönös hitelesítés: mind a kliens, mind a a szerver ellenőrzi a másik felet

- X.509 szabvány - tanusítvány formája
- SSL - meghatározza a tanusítványok cseréjének módját

```
<login-config>
    <auth-method>CLIENT-CERT</auth-method>
    <realm-name>file</realm-name>
</login-config>
```

Olvasnivaló

- Marty Hall: http://www.javapassion.com/j2ee/SSL.pdf
- Marty Hall, http://www.javapassion.com/j2ee/WebSecurityThreats.pdf