Laborator 12: Introducere Spring security (2)

LDAP Authentication

LDAP Lightweight Directory Access Protocol

Protocol care definește modul în care sunt reprezentate datele într-un *directory information service* precum și modul în care infomațiile sunt importate/exportate din/în director într-o rețea IP. La nivel abstract un directory service este o colecție de obiecte organizate ierarhic.

Caracteristici:
write-once-read-many-times service
reprezentarea abstractă a datelor separată de implementare
metode standardizate pentru aceesarea informațiilor

Information model: modul în care sunt reprezentate datele – ierarhie de obiecte.

Entity Entitățile formează o ierarhie de obiecte **Directory Information Tree (DIT)**.
base sau **root** este rădăina arborelui DIT. Fiecare entitate are o entitate părinte și 0 sau mai multe entități fiu. Se poate adăuga în arbore o entitate doar dacă a fost adăugată anterior etitatea părinte.

Fiecare entitate este o instanță a uneia sau mai multor **objectClasses.**
Fiecare entitate instanțiază o clasă **STRUCTURAL** și poate instnția 0 sau mai multe clase **AUXILIARY.** Fiecare objectClass are un nume unic și poate moșteni 0 sau mai multe objectClasses.
objectClass conține 0 sau mai multe atribute. Fiecare atribut are un nume și poate avea una sau mai multe valori (SINGLE-VALUE/MULTI-VALUE).
Unele atribute sunt obligatorii altele sunt opționale (MUST/MAY).

LDAP Data Interchange Files (LDIF) sunt fișiere utilizate pentru inițializarea DIT.
dn identifică unic o entitate și specifică modul în care aceasta va fi adăugată în arbore (calea către fiecare entitate din arbore).
În cazul în care LDAP este utilizat pentru autentificare reprezintă id-ul care va fi utilizat pentru logare.

Atribute uzuale:
dc objectClass: domainComponent any part of a domain name e.g. domain.com, domain or com
sn objectClass: person family name **gn** objectClass:person first name
uid objectClass user id
https://docs.spring.io/spring-security/site/docs/3.0.x/reference/ldap.html

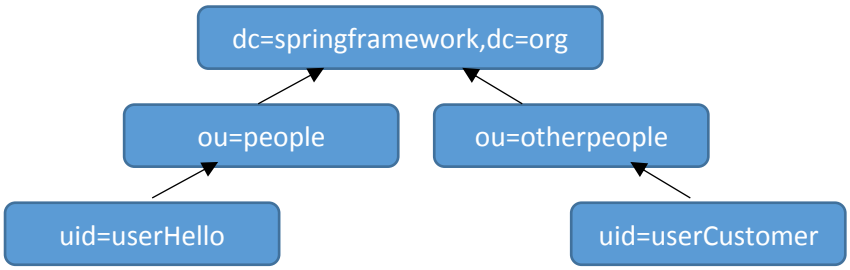
```
dn: dc=springframework,dc=org
objectclass: top
objectclass: domain
objectclass: extensibleObject
dc: springframework

dn: ou=people,dc=springframework,dc=org
objectclass: top
objectclass: organizationalUnit
ou: people

dn: uid=hello,ou=people,dc=springframework,dc=org
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: User Hello
sn: Hello
uid: hello
userPassword: {SHA}fDYHuOYbzxlE6ehQOmYPIfS28/E=
```

1. Importați proiectul LAB12_START și verificați dependențele pentru spring-security și ldap:

```
<dependency>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-security</artifactId>
</dependency>

<dependency>
		<groupId>org.thymeleaf.extras</groupId>
		<artifactId>thymeleaf-extras-springsecurity5</artifactId>
		<version>3.0.4.RELEASE</version>
</dependency>

<dependency>
		<groupId>org.springframework.ldap</groupId>
		<artifactId>spring-ldap-core</artifactId>
		<version>2.3.2.RELEASE</version>
</dependency>

<dependency>
		<groupId>org.springframework.security</groupId>
		<artifactId>spring-security-ldap</artifactId>
		<version>5.1.3.RELEASE</version>
</dependency>
```

```
<dependency>
        <groupId>com.unboundid</groupId>
        <artifactId>unboundid-ldapsdk</artifactId>
        <version>4.0.9</version>
</dependency>
```

2. Creați în WEB-INF users.ldif care va conține următoarele entități:



3. Adăugați în fișierele application.properties configurarea serverului LDAP embeded:

```
spring.ldap.embedded.ldif=classpath:users.ldif
spring.ldap.embedded.base-dn=dc=springframework,dc=org
spring.ldap.embedded.port=8389
```

4. Adăugați clasele apbdoo.laboratorul12.SecurityWebApplicationInitializer și apbdoo.laboratorul12.SecurityConfig

```
public class SecurityWebApplicationInitializer
        extends AbstractSecurityWebApplicationInitializer {
}
```

```
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter
{
    @Override
    protected void configure(AuthenticationManagerBuilder auth)
throws Exception {
        auth.ldapAuthentication()
                .userDnPatterns("uid={0},ou=people")
                .contextSource()
                .root("dc=springframework,dc=org")
          .url("ldap://localhost:8389/dc=springframework,dc=org")
                .and().passwordCompare()
                .passwordEncoder(  new  LdapShaPasswordEncoder())
                .passwordAttribute("userPassword");
    }
}
```

5. Adăugați dependențele:

```
<dependency>
	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-starter-test</artifactId>
</dependency>

<dependency>
	<groupId>org.springframework.security</groupId>
	<artifactId>spring-security-test</artifactId>
	<scope>test</scope>
</dependency>
```

6. Adăugați clasa de test:

```
@SpringBootTest
@AutoConfigureMockMvc
@RunWith(SpringRunner.class)
public class LoginTest {

    @Autowired
    private MockMvc mockMvc;

    @Test
    public void loginValidTest() throws Exception {
        SecurityMockMvcRequestBuilders.FormLoginRequestBuilder
login = formLogin().user("hello").password("abcd123");

mockMvc.perform(login).andExpect(authenticated().withUsername("he
llo"));
    }
}
```

7. Adaugați utilizatorul hello la grupul MANAGER

```
dn: ou=groups,dc=springframework,dc=org
objectclass: top
objectclass: organizationalUnit
ou: groups

dn: cn=managers,ou=groups,dc=springframework,dc=org
objectclass: top
objectclass: groupOfUniqueNames
cn: managers
ou: manager
uniqueMember: uid=hello,ou=people,dc=springframework,dc=org
```

8. În clasa de configurare includeți baza pentru căutarea grupurilor în serverul ldap

```
auth.ldapAuthentication()
            .userDnPatterns("uid={0},ou=people")
            .groupSearchBase("ou=groups")
            .userSearchBase("ou=people")
```

9. Modificați prefixul default care se adaugă rolurilor.

```
.rolePrefix("")
```

10. Configurați parametrul timeout în fișierul application.properties

```
server.servlet.session.timeout=1

echivalent in web.xml
<session-config>
   <session-timeout>20</session-timeout>
</session-config>
```

11. În clasa de configurare stabiliți pagina afișată în cazul în care sesiunea a expirat:

```
protected void configure(HttpSecurity http) throws Exception {
    ….

     http.sessionManagement()
           .maximumSessions(1)
           .expiredUrl("/login?expired");

}
```

12. Adăugați un formular custom pentru login:

```
@GetMapping("/showLogInForm")
   public String showLogInForm(){
      return "login";
}
```

13. Adăugați opțiunea remember me:

```
<p>Remember Me:
        <input type="checkbox" name="remember-me" />
</p>
```

14. În clasa SecurityConfig activați opțiunea remember me (cheia care va fi memorată în cookie împreună cu userId și timpul de expirare):

```
http.rememberMe().
            key("cheiesecreta").tokenValiditySeconds(3600);
```