Integrating ApacheDS with Spring Security

1. Software(s) used:
   1. ApacheDS
   2. Spring Security
2. Install Apache DS
3. The ApacheDS Data Model we used:
4. version: 1

dn: dc=jboss,dc=com

objectclass: top

objectclass: domain

dc: jboss

dn: ou=users,dc=jboss,dc=com

objectClass: organizationalUnit

objectClass: top

ou: users

dn: ou=roles,dc=jboss,dc=com

objectClass: organizationalUnit

objectClass: top

ou: roles

dn: uid=shantanu,ou=users,dc=jboss,dc=com

objectClass: organizationalPerson

objectClass: person

objectClass: inetOrgPerson

objectClass: top

cn: shantanu

sn: banerjee

uid: shantanu

userPassword:: e1NTSEF9dkdHaHVjQVNPaTExWGZBQ3RGZzU5cDZ4dEw5MTdRRlR4NGtSU0E9P

Q==

dn: uid=pavan,ou=users,dc=jboss,dc=com

objectClass: organizationalPerson

objectClass: person

objectClass: inetOrgPerson

objectClass: top

cn: pavan

sn: kumar

uid: pavan

userPassword:: e1NTSEF9dkdHaHVjQVNPaTExWGZBQ3RGZzU5cDZ4dEw5MTdRRlR4NGtSU0E9P

Q==

dn: cn=user,ou=roles,dc=jboss,dc=com

objectClass: groupOfUniqueNames

objectClass: top

cn: user

uniqueMember: uid=pavan,ou=users,dc=jboss,dc=com

dn: cn=admin,ou=roles,dc=jboss,dc=com

objectClass: groupOfUniqueNames

objectClass: top

cn: admin

uniqueMember: uid=pavan,ou=users,dc=jboss,dc=com

uniqueMember: uid=shantanu,ou=users,dc=jboss,dc=com

dn: cn=manager,ou=roles,dc=jboss,dc=com

objectClass: groupOfUniqueNames

objectClass: top

cn: manager

uniqueMember: uid=pavan,ou=users,dc=jboss,dc=com

dn: uid=arun,ou=users,dc=jboss,dc=com

objectClass: organizationalPerson

objectClass: person

objectClass: inetOrgPerson

objectClass: top

cn: arun

sn: kumar

uid: arun

userPassword:: e1NTSEF9dkdHaHVjQVNPaTExWGZBQ3RGZzU5cDZ4dEw5MTdRRlR4NGtSU0E9P

Q==

Spring Application:

**Gradle Dependencies:**

plugins **{** id 'org.springframework.boot' version '2.3.9.RELEASE'  
 id 'io.spring.dependency-management' version '1.0.11.RELEASE'  
 id 'java'  
**}**group = 'com.demo'  
version = '0.0.1-SNAPSHOT'  
sourceCompatibility = '1.8'  
  
repositories **{** mavenCentral()  
**}**dependencies **{** implementation 'org.springframework.boot:spring-boot-starter-security'  
 implementation 'org.springframework.boot:spring-boot-starter-web'  
 *//implementation 'org.springframework.boot:spring-boot-starter-data-ldap'* implementation 'org.springframework.boot:spring-boot-starter-web'  
 implementation 'org.springframework.security:spring-security-ldap'  
 implementation 'org.springframework.boot:spring-boot-starter-jdbc'  
 implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'  
 *//runtimeOnly 'com.oracle.database.jdbc:ojdbc8'* implementation 'org.mariadb.jdbc:mariadb-java-client:2.7.3'  
  
 testImplementation 'org.springframework.boot:spring-boot-starter-test'  
 testImplementation 'org.springframework.security:spring-security-test'  
**}**test **{** useJUnitPlatform()  
**}**

Spring Boot Application:

package com.demo.spring;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
  
@SpringBootApplication  
public class SecureAppApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.*run*(SecureAppApplication.class, args);  
 }  
  
}

Web Controller Class:

package com.demo.spring;  
  
import org.springframework.security.access.annotation.Secured;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestMethod;  
  
import javax.annotation.security.RolesAllowed;  
  
@Controller  
@RequestMapping("app")  
public class DemoController {  
 @RequestMapping(path="mypage",method = RequestMethod.*GET*)  
 public String getPage(){  
 return "securepage";  
 }  
 @RequestMapping(path="adminpage",method = RequestMethod.*GET*)  
 @Secured({"ROLE\_ADMIN"})  
 public String getAdminPage(){  
 return "admin";  
 }  
}

REST Controller:

package com.demo.spring;  
  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
@RequestMapping("services")  
public class MyRestController {  
  
 @GetMapping(path = "greet")  
 public String greet() {  
 return "Welcome to Secured Rest Service";  
 }  
}

Security Configuration:

package com.demo.spring;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.security.config.annotation.SecurityBuilder;  
import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;  
import org.springframework.security.config.annotation.method.configuration.EnableGlobalMethodSecurity;  
import org.springframework.security.config.annotation.web.WebSecurityConfigurer;  
import org.springframework.security.config.annotation.web.builders.HttpSecurity;  
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;  
import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;  
import org.springframework.security.core.userdetails.User;  
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;  
import org.springframework.security.crypto.password.LdapShaPasswordEncoder;  
  
import javax.sql.DataSource;  
  
@Configuration  
@EnableWebSecurity  
@EnableGlobalMethodSecurity(securedEnabled = true,prePostEnabled = true)  
public class SecurityConfig extends WebSecurityConfigurerAdapter {  
 @Autowired  
 DataSource ds;  
  
 @Override  
 protected void configure(HttpSecurity http) throws Exception {  
 http  
 .authorizeRequests()  
 .antMatchers("/app/\*\*")  
 .hasAnyRole("CLIENT","USER","ADMIN")  
 .antMatchers("/all/\*").permitAll()  
 .antMatchers("/services/\*\*").authenticated()  
 .and()  
 .csrf().disable()  
 .httpBasic();  
 *// .formLogin();* }  
*/\*  
 @Autowired  
 public void configureGlobal(AuthenticationManagerBuilder builder) throws Exception{  
 builder.inMemoryAuthentication().withUser("shantanu").password("$2a$10$4J11btA92rm58PRIEiPMEO0s1sYnZslqUXUzAtic3r9ieVrvagkRu").roles("USER");  
 builder.inMemoryAuthentication().withUser("pavan").password("$2a$10$4J11btA92rm58PRIEiPMEO0s1sYnZslqUXUzAtic3r9ieVrvagkRu").roles("CLIENT");  
 builder.inMemoryAuthentication().withUser("mahesh").password("$2a$10$4J11btA92rm58PRIEiPMEO0s1sYnZslqUXUzAtic3r9ieVrvagkRu").roles("ADMIN");  
 }\*/  
  
  
 /\*@Autowired  
 public void configureGlobal(AuthenticationManagerBuilder builder) throws Exception{  
 builder.jdbcAuthentication().dataSource(ds).withDefaultSchema();  
  
 // .usersByUsernameQuery("select username,password,enabled from users where username=?")  
 //.authoritiesByUsernameQuery("select username,authority from authorities where username=?");  
 }\*/* @Autowired  
 public void configureGlobal(AuthenticationManagerBuilder builder) throws Exception{  
 builder.ldapAuthentication()  
 .contextSource()  
 .url("ldap://localhost:10389/dc=jboss,dc=com")  
 .managerDn("uid=admin,ou=system")  
 .managerPassword("secret")  
 .and()  
 .userSearchBase("ou=users")  
 .userSearchFilter("uid={0}")  
 .groupSearchBase("ou=roles")  
 .groupSearchFilter("uniqueMember={0}")  
 .groupRoleAttribute("cn")  
 .passwordCompare()  
 .passwordEncoder(new LdapShaPasswordEncoder())  
 .passwordAttribute("userPassword");  
 System.*out*.println("LDAP initialized...");  
 }  
  
  
 @Bean  
 public BCryptPasswordEncoder encoder(){  
 return new BCryptPasswordEncoder();  
 }  
  
}

Apache DS Data Model:

