AntMachers in spring boot security

In

=> Spring boot seurity App every URL (nothing but request path of handler method) must be

configured with security using permitAll()(no authentication and no authorization), authenticatied() (only authentication) hasRole(), hasAnyRole() (Authetication +Authorization).. For this we need to use AntMatcher concept

in SecurityConfig class.

case1::

AntMatcher for Multipath @Controller

```
@RequestMapping("/customer") //global path
public class CustomerController{
}
@GetMapping("/register")
public String registerCustomer(){.....}
@GetMapping("/delete")
public String deleteCustomer(){.....}
@GetMapping("/update")
public String updateCustomer(){.....}
```

=> To match with mutlilevel path we can give </path> like /customer** in AntMatcher note:: multiple level path is like /customer/register, /customer/delete, /customer/update, /customer/register/abc, /customer/update/type**

```
.antMatchers("/customer**).hasRole("MANAGER"); // Only "MANAGER" role authenticated users can access web pages whose URLs starts with /customer and contains multi path
```

case2: AntMatcher for

singleLevel path

```
@Controller
public class CustomerController
@PostMapping("/registerCustomer")
public String registerCustomer(){
}
---
@PostMapping("/registerProduct")
public String registerProduct(){
}
...
@PostMapping("/registerFaculty")
public String registerFaculty(){
}
```

...

}

we can give AntMatcher using </path>* pattern..

.antMatchers("/register*").hasAnyRole("MANAGER","CUSTOMER")

matchers with /registerCustomer, /registerProduct, /registerFaculty urls

case3: Multiple URLs can be given in single AntMacher expression

version1:

.antMatchers("/save").hasRole("MANAGER")

.antMatchers("/update").hasRole("MANAGER")

.antMatchers("/delete").hasRole("MANAGER")

version2 (Improved code of version1)

.ant Matchers("/save","/update","/delete").hasRole("MANAGER")

case4: Left over request urls can be identified and mapped using .anyRequest() expression.

Let assume we are having multuple request urls/paths as shown below

"/save", "/update", "/delete","/report","/upload",

"/download", "/paging","/info","/aboutUs"

.antMatchers("/save","/update").hasRole("CUSTOMER")

.antMatchers("/report","/upload","/download").hasRole("MANAGER")

.anyRequest().authenticated(); // represents the left over urls like "/paging","/aboutUs","/info"

used

=>{noop}<pwd> indicates Password is not encoded indirectly it says "NoopEncoder" is to econde the passowrd. =>Initial spring security used allow not encoded passwords.. Later it stopped allowing them .. So to pass Non-encoded passwords we need use {noop}

=> We can nsdifferent Encoders like "BCryptEndcoder" and etc.. to encode the passowrds..

"Base64Encoder" "SHA512Encoder" "MD5Encoder"

=> If do not to use {noop} expression based "NoopPasswordEncoder" then we need to pass encoded passwords as shown below.

step1) take sepeate App to get Encoded passwords.

package com.nt.encrypt;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

public class PasswordEncoder {

public static void main(String[] args) {

BCryptPasswordEncoder encoder=new BCryptPasswordEncoder();

String pwd1-encoder.encode("rani");

String pwd2-encoder.encode("hyd");

System.out.println(pwd); System.out.println(pwd2);

}

How to lock the use while working with InMemory DB?

auth.inMemoryAuthentication().passwordEncoder(new BCryptPassword Encoder()).withUser("raja").password("$2a$10

$/JADWLU6mFbf5F.ulzLDduBrLcr7NJEVOM6aUYnk5UcJeV2EX5WAe").roles("CUSTOMER").accountLocked(true);

step2) Run the above App and get Encoded passwords and use them in 1st configure (-) method of Securityconfig class

@Override

public void configure (AuthenticationManagerBuilder auth) throws Exception {

auth.inMemoryAuthentication().passwordEncoder(new BCryptPasswordEncoder()).withUser("raja").password("$2a$10$irWQwKOV6vm9Ksf7b11Ttu1RrjrJUrGQK4Pah8FGj8JvngJqjKKqe").roles("CUSTOMER");

auth.inMemoryAuthentication(). passwordEncoder(new BCryptPasswordEncoder()). withUser("ramesh"). password("$2a$10$SgFbtmR9u3SXkvR7rQrdbuJYhtJuHNqp|B2uoVS.sh EtSGkb2ibmS").roles("MANAGER");

}

note: always recomanded to work with encoded password to manage the passwords with strong encyption.

Do i need to need encode the passwords manually as shown above in the real projects?

having

ans) Definitely not.. As part of user registration logic we include our choice encoder to get encoded password for the given password they will be saved db table..

to

Working with jdbcAuthentication that uses spring JDBC based DB s/w as AutheticationInfo Provider

==========

step1) maker that following db tables are avaiable in any db s/w like oracle

CREATE TABLE "SYSTEM"."USERS"

("UNAME" VARCHAR2(20 BYTE) NOT NULL ENABLE,

Parent db table

"PWD" VARCHAR2(70 BYTE),

"STATUS" NUMBER(1,0),

CONSTRAINT "USERS_PK" PRIMARY KEY ("UNAME"));

USERS

CREATE TABLE "SYSTEM"."USER_ROLES"

("ROLE" VARCHAR2(20 BYTE),

"UNAME" VARCHAR2(20 BYTE),

CONSTRAINT "FK1" FOREIGN KEY ("UNAME")

FK colunin REFERENCES "SYSTEM"."USERS" ("UNAME") ENABLE));

(parent db table)

Columns Data Model | Constraints Grants Statistics | Triggers Flashback | Dependencies | Details Partitions Indexes SQL

+ 74X

(PK) UNAME

1 raja

2 ramen

Sort.. Filter:

PWD

$2a$10$irWQwKOV6vm9Ksf7b11TtulRrjrJUGQK4 Pah8FGj8JvngJqjKKqe

$2a$10$SgFbtmR9u3SXkvR7rQrdbuJYht JuHNqp1B2uoVS.shEtSGkb2ibmS

USER_ROLES

(child db table

Columns Data Moder constraints Grants Statistics Triggers | Flashbac

HE

Sort..

ROLE

1 MANAGER

2 CUSTOMER

3 CUSTOMER

UNAME ramesh

(FK)

raja

ramesh

child

db table

=>Taking user details and

roles details in single db table

is bad pratice.. This approach

do not support one user having multiple roles

so prefer two db tables having FK relationship

Table names and col names

not fixed beoz SQL Queries in Security Config class seperately

we are going place

step2) add spring jdbc, oracle driver staters to project as additional staters.. Right click project --->spring -->add starters --->select JDBC API,Oracle

step3) add jdbc properties in application.properties file in application.properties

spring.datasource.driver-class-name-oracle.jdbc.driver.Oracle Driver
spring.datasource.url=jdbc:oracle:thin:@localhost:1521:xe

spring.datasource.username=system

spring.datasource.password=manager

step4) Inject DataSource object to SecurityConfig class

```
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter { @Autowired
private DataSource ds;
step5) write the following code in 1st configure(-) method to enable jdbc Authentication
(Collect from mvn repository.com)
<!-- https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-jdbc --> <dependency>
<groupId>org.springframework.boot</groupId> <artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>
<!-- https://mvnrepository.com/artifact/com.oracle.database.jdbc/ojdbc11 --> <dependency>
<groupId>com.oracle.database.jdbc</groupId>
<artifactId>ojdbc11</artifactId>
</dependency>
@Override
public void configure(AuthenticationManagerBuilder auth) throws Exception {
auth.jdbcAuthentication().dataSource(ds).passwordEncoder(new BCryptPasswordEncoder())
.usersByUsernameQuery("SELECT UNAME,PWD,STATUS FROM USERS WHERE UNAME=?") //for
autheentication .authoritiesByUsernameQuery("SELECT UNAME,ROLE FROM USER_ROLES WHERE
UNAME=?"); //for authorization
}
@Override
public void configure(HttpSecurity http) throws Exception {
}
http.authorizeRequests().antMatchers("/").permitAll()
.antMatchers("/offers").authenticated()
.antMatchers("/balance").hasAnyAuthority("CLERK","MANAGER")
.antMatchers("/loan").hasAuthority("MANAGER")
.anyRequest().authenticated()
.and().formLogin()
.and().rememberMe()
note:: do not use hasAnyRole(-) or hasRole(-) methods here
.and().logout().logout Request Matcher(new AntPathRequestMatcher("/signout"))
//.and().logout()
//.and().httpBasic() // Enables BASIC Authentication - makes the browser generating the dialogbox asking for
username, password.
//cfg errror page for 403 error (authorization error)
step6) Run the Application
.and().exceptionHandling().accessDenied Page("/denied")
```

```
.and().sessionManagement().maximumSessions (2).maxSessions PreventsLogin(true);
```