CS544 EA

Applications

Spring Security: Authentication Providers

Plain Text

- So far we've used plain text: **bad for security**

```
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
     auth.inMemoryAuthentication()
     .withUser("admin").password("{noop}123").roles("USER","ADMIN").and()
     .withUser("user").password("{noop}bla").roles("USER");
}
```

```
<authentication-manager>
    <authentication-provider>
        <user-service>
            <user name="test" password="{noop}123" authorities="ROLE_USER, ROLE_ADMIN" />
            <user name="bob" password="{noop}bobiscool" authorities="ROLE_USER" />
        </user-service>
    </authentication-provider>
</authentication-manager>
```

Password Encoder

- Super important:
  - **Never store plain text** (no reading of pwds!)
  - Basic hashing isn't that great either

```
@Bean
public BCryptPasswordEncoder passwordEncoder() {
    return new BCryptPasswordEncoder();
}
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
      auth.inMemoryAuthentication()
      .withUser("jimi").password("{bcrypt}d7e6351eaa13189a5a3641bab846c8e8c69ba39f").roles("ADMIN","USER").and()
      .withUser("bob").password("{bcrypt}4e7421b1b8765d8f9406d87e7cc6aa784c4ab97f").roles("USER");
}
```

```
<sec:authentication-manager>
  <sec:authentication-provider>
    <sec:password-encoder hash="bcrypt"/>
    <sec:user-service>
      <sec:user name="jimi" password="{bcrypt}d7e6351eaa13189a5a3641bab846c8e8c69ba39f" authorities="ROLE_USER, ROLE_ADMIN" />
      <sec:user name="bob" password="{bcrypt}4e7421b1b8765d8f9406d87e7cc6aa784c4ab97f" authorities="ROLE_USER" />
    </sec:user-service>
  </sec:authentication-provider>
</sec:authentication-manager>
```

Can be: md4, md5, sha, sha-256, bcrypt

Bcrypt is recommended as it also auto-salts

JDBC Authenticator

```
@Autowired
private DataSource dataSource;

@Override
public void configure(AuthenticationManagerBuilder auth) throws Exception {
      auth.jdbcAuthentication()
      .dataSource(dataSource)
      .withDefaultSchema()
}
```

```
<sec:authentication-manager>
  <sec:authentication-provider>
    <sec:jdbc-user-service data-source-ref="dataSource" />
  </sec:authentication-provider>
</sec:authentication-manager>

<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
  <property name="driverClassName" value="com.mysql.jdbc.Driver"/>
  <property name="url" value="jdbc:mysql://localhost/cs544"/>
  <property name="username" value="root"/>
  <property name="password" value="root"/>
</bean>
```

Standard Authentication Tables

JDBC authentication expects the following tables:

```
create table users(
    username varchar_ignorecase(50) not null primary key,
    password varchar_ignorecase(50) not null,
    enabled boolean not null
);
create table authorities (
    username varchar_ignorecase(50) not null,
    authority varchar_ignorecase(50) not null,
    constraint fk_authorities_users foreign key(username) references users(username)
);
create unique index ix_auth_username on authorities (username,authority);
```

Values could be inserted like so:

```
Insert into users values(“test”, “{bcrypt}d7e6351eaa13189a5a3641bab846c8e8c69ba39f”, 1”);
Insert into users values(“bob”, “{bcrypt}4e7421b1b8765d8f9406d87e7cc6aa784c4ab97f”, 1”);
Insert into authorities values(“test”, “ROLE_USER”);
Insert into authorities values(“test”, “ROLE_ADMIN”);
Insert into authorities values(“bob”, “ROLE_USER”);
```

Custom Tables / Queries

```
@Override
public void configure(AuthenticationManagerBuilder auth) throws Exception {
    auth.jdbcAuthentication()
    .dataSource(dataSource)
    .usersByUsernameQuery("select username,password, enabled from users where username=?")
    .authoritiesByUsernameQuery("select u.username, ur.authority from users u, user_roles ur where u.user_id = ur.user_id " +
        " and u.username =?");
}
```

```
<sec:authentication-manager>
  <sec:authentication-provider>
    <sec:jdbc-user-service data-source-ref="dataSource"
      users-by-username-query="select username,password, enabled from users where username=?"
      authorities-by-username-query="select u.username, ur.authority from users u, user_roles ur where u.user_id = ur.user_id
        and u.username =?" />
  </sec:authentication-provider>
</sec:authentication-manager>
```

Custom Authentication Provider

```java
public class CustomAuthenticationProvider implements AuthenticationProvider {
  @Override
  public Authentication authenticate(Authentication authentication) throws AuthenticationException {
    String name = authentication.getName();
    String password = authentication.getCredentials().toString();
    if (name.equals("test") && password.equals("123")) {
      List<GrantedAuthority> grantedAuths = new ArrayList<>();
      grantedAuths.add(new SimpleGrantedAuthority("ROLE_USER"));
      Authentication auth = new UsernamePasswordAuthenticationToken(name, password, grantedAuths);
      return auth;
    } else {
      return null;
    }
  }
  @Override
  public boolean supports(Class<?> authentication) {
    return authentication.equals(UsernamePasswordAuthenticationToken.class);
  }
}
```

```java
@Bean
public CustomAuthenticationProvider customAuthenticationProvider() {
    return new CustomAuthenticationProvider();
}
```

```xml
<sec:authentication-manager>
   <sec:authentication-provider ref="customAuthenticationProvider"/>
</sec:authentication-manager>
```

Multiple Authentication Providers

- Spring will try each one, in the order found

```
<sec:authentication-manager>

    <sec:authentication-provider>
        <sec:user-service>
            <sec:user name="test" password="{noop}123" authorities="ROLE_USER, ROLE_ADMIN" />
            <sec:user name="bob" password="{noop}bobiscool" authorities="ROLE_USER" />
        </sec:user-service>
    </sec:authentication-provider>

    <sec:authentication-provider>
        <sec:jdbc-user-service data-source-ref="dataSource" />
    </sec:authentication-provider>

    <sec:authentication-provider ref="customAuthenticationProvider" />
</sec:authentication-manager>
```

I have not tested this yet for Java Config