Security in Web Applications

Piotr Łochowski

November 25, 2015

Agenda

- Threats
- Security - under the hood
- Authentication
- Authorization
  - Roles
  - Business rules
  - ACL
- Summary

Major Threats

- Who is who?
- Phishing
- Malware (malicious software)
- Identity Theft

How do we secure our resources?

Authorization vs. Authentication

Authentication. Which way to go?

- Form login (login/password) authentication
- Basic and Digest authentication
- Remember-Me (cookie) authentication
- LDAP authentication
- Kerberos, Active Directory authentication
- JAAS (Java Authentication and Authorization)
- CAS (Central Authentication Service)
- X.509
- OpenID2, OAuth2
- Google. Facebook, Twitter, Linkedin, Github, Tripit, etc. (http://projects.spring.io/spring-social/)

@EnableWebSecurity

WebSecurityConfigurerAdapter

```
@Override
protected void configure(HttpSecurity http)
     throws Exception {
  http
    .authorizeRequests()
      .antMatchers("/", "/home").permitAll()
      .anyRequest().authenticated()
      .and()
    .formLogin()
      .loginPage("/login")
      .permitAll()
      .and()
    .logout()
      .permitAll();
}
```

AuthenticationManagerBuilder

```
@Autowired
public void configureGlobal(
              AuthenticationManagerBuilder auth)
              throws Exception {
  auth
    .inMemoryAuthentication()
      .withUser("user")
      .password("password")
      .roles("USER");
}
```

Demo

Login Form

Authorization models

- Roles
- Business rules
- ACLs

I'm in role. I act as.

Roles are focused on very simple functionality.

Allows granting permissions.

User usually has more than one role.

But who am I, what can I do?

```
Object principal = SecurityContextHolder
                        .getContext()
                        .getAuthentication()
                        .getPrincipal();
UserDetails userDetails = (UserDetails) principal;
```

Expression-Based Access Control

- hasRole([role])
- hasAnyRole([role1,role2])
- principal
- authentication
- permitAll
- denyAll
- isAnonymous()
- isRememberMe()
- isAuthenticated()
- isFullyAuthenticated()

Demo

Role based permission

```
@EnableGlobalMethodSecurity(prePostEnabled = true)
@PreAuthorize("hasRole('MANAGER')")
```

Business rules is what you are looking for

I can view photos of may friends only.

I can comment photos of my close friends.

Premium user can rate photo.

Business Rules

```
@PreAuthorize(
    "#bookmark.owner.name == authentication.name")
void doSomething(Bookmark bookmark);
```

Business Rules 2

```
@Service("bookmarkService")
class BookmarkService

@PreAuthorize("@alertAuthorityService.canView(#queryId)")
BookmarkView load(@PathVariable("id") Long queryId)
```

ACL is very tricky

An ACL specifies which users or system processes are granted access to objects, as well as what operations are allowed on given objects.

I will save password in plain text, just for now

- Plaintext
- MD5
- SHA-xxx
- bcrypt/scrypt

Password Encoder

```
@Bean
public PasswordEncoder passwordEncoder() {
 return new BCryptPasswordEncoder();
}

@Bean
public DaoAuthenticationProvider authenticationProvider() {
  DaoAuthenticationProvider authenticationProvider =
                          new DaoAuthenticationProvider();
  ...
  authenticationProvider.
       setPasswordEncoder(passwordEncoder());
  return authenticationProvider;
}
```

I can do better. Two phase authentication!

- You know (Your mother's maiden name)
- You possess (ID card, Driving License)
- You are (fingerprint)

OTP - One Time Password

- You know (Your mother's maiden name)
- You possess (ID card, Driving License)
- You are (fingerprint)

Remember to

- Using HTTPS including login form
- Do not send password in plain text, use password encoder instead
- ... but not MD5!
- Use two face authentication if possible
- Think twice before you incorporate ACL