Spring Security ve Entegrasyon Testleri

Spring Security ve Entegrasyon Testleri

- Spring ApplicationContext'in de yer aldığı **entegrasyon birim testleri**nde servis metot çağrılarının yetkilendirilmeleri de kontrol edilebilir
- Bunun için öncelikle test metodu çalışmadan evvel kimliklendirme adımının gerçekleşmiş ve SecurityContext'de uygun rollere sahip **geçerli bir Authentication token**'ın mevcut olması gerekir

Spring Security ve Entegrasyon Testleri

- Spring Security için herhangi bir biçimde **SecurityContext**'de geçerli bir **Authentication** token konması yeterlidir
- Bu nesnenin kim tarafından **nasıl konulduğu önemli değil**dir
- Önemli olan yetkilendirme adımından evvel **bu işlemin bir biçimde yapılmış olması**dır
- SecurityContext'in içeriğinin uygulama içerisinde **doğrudan programatik olarak set edilmesi** mümkündür

Spring Security ve Entegrasyon Testleri

- **TestingAuthenticationToken** ile valid bir authentication token oluşturulabilir
- Bu token doğrudan **SecurityContext'e** set edilebilir
- Bu token'ın doğrulanması için sistemde **TestingAuthenticationProvider**'ın authentication provider olarak tanıtılmış olması gerekir

Spring Security ve Entegrasyon Testleri

AuthenticationManager bean'ine Testing AuthenticationProvider da register edilmelidir

```
<beans>

   <security:authentication-manager>
   ...
      <security:authentication-provider
         ref="testingAuthenticationProvider"/>

   </security:authentication-manager>

   <bean id="testingAuthenticationProvider"
class="org.springframework.security.authentication.T
estingAuthenticationProvider"/>

</beans>
```

Spring Security ve Entegrasyon Testleri

```
@Before
public void setUp() {
    TestingAuthenticationToken auth =
        new TestingAuthenticationToken("myuser", "secret", "ROLE_USER");
    SecurityContextHolder.getContext().setAuthentication(auth);
}

@Test
public void testSecureMethodCalls() {
    businessService.secureMethod();
}

@Test
public void tearDown() {
    SecurityContextHolder.clearContext();
}
```

Spring Security Test Kütüphanesi

- Spring Security 5 ile birlikte entegrasyon testlerinde **kullanıcı ve rol bilgilerinin SecurityContext'e kolayca populate edilmesi**ni sağlayan bir kabiliyet gelmiştir

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-test</artifactId>
    <version>5.0.9.RELEASE</version>
</dependency>
```

@WithMockUser Kullanımı

```
@RunWith(SpringJUnit4Runner.class)
@ContextConfiguration
@SecurityTestExecutionListeners
public class SecurityTests {
    @Configuration
    static class Config {
    }

    @Autowired
    private BusinessService businessService;

    @Test
    @WithMockUser(username="myuser",password="secret",roles="USER")
    public void testSecureMethodCalls() {
        businessService.secureMethod();
    }
}
```

@WithMockUser, @WithAnonymousUser, @WithUserDetails anotasyonlarını devreye sokar

Verilen parametrelerle UsernamePasswordauthenticationToken oluşturarak SecurityContext'i populate eder

@WithAnonymousUser

```
@RunWith(SpringJUnit4Runner.class)
@ContextConfiguration
@SecurityTestExecutionListeners
public class SecurityTests {
    @Configuration
    static class Config {
    }

    @Autowired
    private BusinessService businessService;

    @Test
    @WithAnonymousUser
    public void testSecureMethodCalls() {
        businessService.secureMethod();
    }
}
```

AnonymousAuthenticationToken oluşturarak SecurityContext'i populate eder

@WithUserDetails

```
@RunWith(SpringJUnit4Runner.class)
@ContextConfiguration
@SecurityTestExecutionListeners
public class SecurityTests {
    @Configuration
    static class Config {
    }

    @Autowired
    private BusinessService businessService;

    @Test
    @WithUserDetails(value="user1")
    public void testSecureMethodCalls() {
        businessService.secureMethod();
    }
}
```

ApplicationContext'de tanımlı UserDetailsService bean'ini kullanarak verilen username'e karşılık gelen UserDetails'i yükleyerek UsernamePasswordAuthenticationToken oluşturur ve SecurityContext'i bununla populate eder

İstenirse userDetailsServiceBeanName attribute ile farklı bir UserDetailsService bean'i kullanması da sağlanabilir

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com