**SPRING SECURITY:**

**Autenticazione:**

Spring security supporta i seguenti standard per autenticazione

HTTP BASIC authentication headers (an IETF RFC-based standard)  
HTTP Digest authentication headers (an IETF RFC-based standard)  
HTTP X.509 client certificate exchange (an IETF RFC-based standard)  
Form-based authentication (for simple user interface needs)  
OpenID authentication  
Transparent authentication context propagation for Remote Method Invocation (RMI) and HttpInvoker (a Spring remoting protocol)  
Automatic "remember-me" authentication (so you can tick a box to avoid re-authentication for a predetermined period of time)  
Anonymous authentication (allowing every unauthenticated call to automatically assume a particular security identity)  
Java Authentication and Authorization Service (JAAS)  
JEE container autentication (so you can still use Container Managed Authentication if desired)  
Single Sign On libs

**Maven:**

Scaricare bom invece che pom in modo da avere tutte le dipendenze a portata di mano.

**Moduli principali: (slide visuale)  
  
  
  
  
  
  
  
  
  
  
  
  
  
DETTAGLIO (1 SLIDE PER CORE, GLI ALTRI CONDIVISI)**

**Core - spring-security-core.jar**

Contains core authentication and access-contol classes and interfaces, remoting support and basic provisioning APIs. Required by any application which uses Spring Security. Supports standalone applications, remote clients, method (service layer) security and JDBC user provisioning. Contains the top-level packages:

org.springframework.security.core

org.springframework.security.access

org.springframework.security.authentication

org.springframework.security.provisioning

**Remoting - spring-security-remoting.jar**

Provides intergration with Spring Remoting. You don’t need this unless you are writing a remote client which uses Spring Remoting. The main package is org.springframework.security.remoting.

**Web - spring-security-web.jar**

Contains filters and related web-security infrastructure code. Anything with a servlet API dependency. You’ll need it if you require Spring Security web authentication services and URL-based access-control. The main package is org.springframework.security.web.

**Config - spring-security-config.jar**

Contains the security namespace parsing code. You need it if you are using the Spring Security XML namespace for configuration. The main package is org.springframework.security.config. None of the classes are intended for direct use in an application.

**LDAP - spring-security-ldap.jar**

LDAP authentication and provisioning code. Required if you need to use LDAP authentication or manage LDAP user entries. The top-level package is org.springframework.security.ldap.

**ACL - spring-security-acl.jar**

Specialized domain object ACL implementation. Used to apply security to specific domain object instances within your application. The top-level package is org.springframework.security.acls.

**CAS - spring-security-cas.jar**

Spring Security’s CAS client integration. If you want to use Spring Security web authentication with a CAS single sign-on server. The top-level package is org.springframework.security.cas.

**OpenID - spring-security-openid.jar**

OpenID web authentication support. Used to authenticate users against an external OpenID server. org.springframework.security.openid. Requires OpenID4Java.

**Demo HelloSecurity (web app)**Risorsa Accessibile solo per utenti autenticati.

1. AFU
   1. Login
   2. Logout
   3. Credenziali
   4. Ruoli
   5. Risorse
2. Dichiarativa:
   1. Web.xml
   2. SecurityFilters
   3. Security-context.xml
3. Da javaConfig
   1. @EnableWebSecurity
   2. AbstractSecurityWebApplicationInitializer
   3. WebSecurityConfigurerAdapter

**Lab:**Aggiungere utenti, ruoli, risorse con vincoli

**Demo SS-200-UrlProtection (web app)**

Obiettivi: espressioni http pattern ed interceptors

Starter:  
1) Controllers accessibili per ruolo (uno per ruolo)  
2) Credenziali ed authorities static (da config)  
  
Tutorial:  
1) Protezione percorsi (Controller, jsp, static res)

**Demo: AuthenticationProviders (modi identificazione): (UI)**

1. Memoria
2. JDBC
3. OpenId
4. Custom
5. Ldap

**Demo Method validation:**

1. Spring annotations
2. Spel Security
3. JSR-250 Annotations
4. Accesso UserInRole e logica per utente
5. Interceptors
6. AspectJ pointcuts

**Demo Profile jsp security tags**

1. Declaring the Taglib
2. The authorize Tag
3. Disabling Tag Authorization for Testing
4. The authentication Tag
5. The accesscontrollist Tag
6. The csrfInput Tag
7. The csrfMetaTags Tag

**Demo Servizi REST:**

1. Autenticazione Certificato HTTP X.509
2. Https

**Demo Utest:**

1. Testing Method Security
2. Security Test Setup
3. @WithMockUser
4. @WithUserDetails
5. @WithSecurityContext

**Architettura: Overview**

1. Oggetti Core
2. Servizi
3. Ciclo di vita richiesta con SS

**Oggetti Core:**

**Servizi**