

Spring Security 3.1



[Spring Security 3.1_ 下载链接1](#)

著者:Robert Winch

出版者:Packt Publishing Ltd

出版时间:2012-12

装帧:

isbn:9781849518260

Secure your web applications from hackers with this step-by-step guide

Learn to leverage the power of Spring Security to keep intruders at bay through simple examples that illustrate real world problems

Each sample demonstrates key concepts allowing you to build your knowledge of the architecture in a practical and incremental way

Filled with samples that clearly illustrate how to integrate with the technologies and frameworks of your choice

In Detail

Knowing that experienced hackers are itching to test your skills makes security one of the most difficult and high-pressure concerns of creating an application. The complexity of properly securing an application is compounded when you must also integrate this factor with existing code, new technologies, and other frameworks. Use this book to easily secure your Java application with the tried and trusted Spring Security framework, a powerful and highly customizable authentication and access-control framework.

"Spring Security 3.1" is an incremental guide that will teach you how to protect your application from malicious users. You will learn how to cleanly integrate Spring Security into your application using the latest technologies and frameworks with the help of detailed examples.

This book is centred around a security audit of an insecure application and then modifying the sample to resolve the issues found in the audit.

The book starts by integrating a variety of authentication mechanisms. It then demonstrates how to properly restrict access to your application. It concludes with tips on integrating with some of the more popular web frameworks. An example of how Spring Security defends against session fixation, moves into concurrency control, and how you can utilize session management for administrative functions is also included.

"Spring Security 3.1" will ensure that integrating with Spring Security is seamless from start to finish.

What you will learn from this book

Understand common security vulnerabilities and how to resolve them

Implement authentication and authorization

Learn to utilize existing corporate infrastructure such as LDAP, Active Directory, Kerberos, and CAS

Integrate with popular frameworks such as Spring, JSF, GWT, Maven, and Spring Roo

Architect solutions that leverage the full power of Spring Security while remaining loosely coupled

Implement common scenarios such as supporting existing user stores, user sign up, and supporting AJAX requests

Approach

This practical step-by-step tutorial has plenty of example code coupled with the necessary screenshots and clear narration so that grasping content is made easier and

quicker.

Who this book is written for

This book is intended for Java web developers and assumes a basic understanding of creating Java web applications, XML, and the Spring Framework. You are not assumed to have any previous experience with Spring Security.

作者介绍:

Robert Winch

Robert Winch is currently a Senior Software Engineer at VMware and is the project lead of the Spring Security framework. In the past he has worked as a Software Architect at Cerner, the largest provider of electronic medical systems in the US. Throughout his career he has developed hands on experience in integrating Spring Security with an array of security standards (i.e. LDAP, SAML, CAS, OAuth, etc). Before he was employed at Cerner, he worked as an independent web contractor in proteomics research at Loyola University, Chicago, and on the Globus Toolkit at Argonne National Laboratory.

Peter Mularien

Peter Mularien is an experienced software architect and engineer, and the author of the book Spring Security 3, Packt Publishing. Peter currently works for a large financial services company and has over 12 years consulting and product experience in Java, Spring, Oracle, and many other enterprise technologies. He is also the reviewer of this book.

目录: Table of Contents

Preface

Chapter 1: Anatomy of an Unsafe Application

Chapter 2: Getting Started with Spring Security

Chapter 3: Custom Authentication

Chapter 4: JDBC-based Authentication

Chapter 5: LDAP Directory Services

Chapter 6: Remember-me Services

Chapter 7: Client Certificate Authentication

Chapter 8: Opening up to OpenID

Chapter 9: Single Sign-on with Central Authentication Service

Chapter 10: Fine-grained Access Control

Chapter 11: Access Control Lists

Chapter 12: Custom Authorization

Chapter 13: Session Management

Chapter 14: Integrating with Other Frameworks

Chapter 15: Migration to Spring Security 3.1

Appendix: Additional Reference Material

Index

Preface

Up

Chapter 1: Anatomy of an Unsafe Application

Security audit

About the sample application

The JBCP calendar application architecture
Application technology
Reviewing the audit results
Authentication
Authorization
Database credential security
Sensitive information
Transport-level protection
Using Spring Security 3.1 to address security concerns
Why Spring Security
Summary
Up
Chapter 2: Getting Started with Spring Security
Hello Spring Security
Importing the sample application
Updating your dependencies
Using Spring 3.1 and Spring Security 3.1
Implementing a Spring Security XML configuration file
Updating your web.xml file
ContextLoaderListener
ContextLoaderListener versus DispatcherServlet
springSecurityFilterChain
DelegatingFilterProxy
FilterChainProxy
Running a secured application
Common problems
A little bit of polish
Customizing login
Configuring logout
The page isn't redirecting properly
Basic role-based authorization
Expression-based authorization
Conditionally displaying authentication information
Customizing the behavior after login
Summary
Up
Chapter 3: Custom Authentication
JBCP Calendar architecture
CalendarUser
Event
CalendarService
UserContext
SpringSecurityUserContext
Logging in new users using SecurityContextHolder
Managing users in Spring Security
Logging in a new user to an application
Updating SignupController
Creating a custom UserDetailsService object
CalendarUserDetailsService
Configuring UserDetailsService
Removing references to UserDetailsManager
CalendarUserDetails
SpringSecurityUserContext simplifications
Displaying custom user attributes

Creating a custom AuthenticationProvider object
CalendarUserAuthenticationProvider
Configuring CalendarUserAuthenticationProvider
Authenticating with different parameters
DomainUsernamePasswordAuthenticationToken
Updating CalendarUserAuthenticationProvider
Adding domain to the login page
DomainUsernamePasswordAuthenticationFilter
Updating our configuration
Which authentication method to use
Summary
Up

Chapter 4: JDBC-based Authentication
Using Spring Security's default JDBC authentication
Required dependencies
Using the H2 database
Provided JDBC scripts
Configuring the H2-embedded database
Configuring JDBC UserDetailsManager
Spring Security's default user schema
Defining users
Defining user authorities
UserDetailsManager
What other features does UserDetailsManager provide out of the box
Group-based access control
Configuring group-based access control
Configuring JdbcUserDetailsManager to use groups
Utilize the GBAC JDBC scripts
Group-based schema
Group authority mappings
Support for a custom schema
Determining the correct JDBC SQL queries
Updating the SQL scripts that are loaded
CalendarUser authority SQL
Insert custom authorities
Configuring the JdbcUserDetailsManager to use custom SQL queries
Configuring secure passwords
PasswordEncoder
Configuring password encoding
Configuring the PasswordEncoder
Making Spring Security aware of the PasswordEncoder
Hashing the stored passwords
Hashing a new user's passwords
Not quite secure
Would you like some salt with that password
Using salt in Spring Security
Summary
Up

Chapter 5: LDAP Directory Services
Understanding LDAP
LDAP
Common LDAP attribute names
Updating our dependencies
Configuring embedded LDAP integration

Configuring an LDAP server reference
Enabling the LDAP AuthenticationProviderNext interface
Troubleshooting embedded LDAP
Understanding how Spring LDAP authentication works
Authenticating user credentials
Demonstrating authentication with Apache Directory Studio
Binding anonymously to LDAP
Searching for the user
Binding as a user to LDAP
Determining user role membership
Determining roles with Apache Directory Studio
Mapping additional attributes of UserDetails
Advanced LDAP configuration
Sample JBCP LDAP users
Password comparison versus bind authentication
Configuring basic password comparison
LDAP password encoding and storage
The drawbacks of a password comparison authenticator
Configuring UserDetailsContextMapper
Implicit configuration of UserDetailsContextMapper
Viewing additional user details
Using an alternate password attribute
Using LDAP as UserDetailsService
Configuring LdapUserDetailsService
Updating AccountController to use LdapUserDetailsService
Integrating with an external LDAP server
Explicit LDAP bean configuration
Configuring an external LDAP server reference
Configuring LdapAuthenticationProvider
Delegating role discovery to UserDetailsService
Integrating with Microsoft Active Directory via LDAP
Built-In Active Directory support in Spring Security 3.1
Summary
Up
Chapter 6: Remember-me Services
What is remember-me
Dependencies
The token-based remember-me feature
Configuring the token-based remember-me feature
How the token-based remember-me feature works
MD5
Remember-me signature
Token-based remember-me configuration directives
Is remember-me secure
Authorization rules for remember-me
Persistent remember-me
Using the persistent-based remember-me feature
Adding SQL to create the remember-me schema
Initializing the data source with the remember-me schema
Configuring the persistent-based remember-me feature
How does the persistent-based remember-me feature work
Are database-backed persistent tokens more secure
Cleaning up the expired remember-me sessions
Remember-me architecture

- Remember-me and the user lifecycle
- Restricting the remember-me feature to an IP address
- Custom cookie and HTTP parameter names
- Summary
- Up
- Chapter 7: Client Certificate Authentication
 - How client certificate authentication works
 - Setting up client certificate authentication infrastructure
 - Understanding the purpose of a public key infrastructure
 - Creating a client certificate key pair
 - Configuring the Tomcat trust store
 - Importing the certificate key pair into a browser
 - Using Firefox
 - Using Chrome
 - Using Internet Explorer
 - Wrapping up testing
 - Troubleshooting client certificate authentication
 - Configuring client certificate authentication in Spring Security
 - Configuring client certificate authentication using the security namespace
 - How Spring Security uses certificate information
 - How Spring Security certificate authentication works
 - Handling unauthenticated requests with AuthenticationEntryPoint
 - Supporting dual-mode authentication
 - Configuring client certificate authentication using Spring Beans
 - Additional capabilities of bean-based configuration
 - Considerations when implementing Client Certificate authentication
 - Summary
 - Up
- Chapter 8: Opening up to OpenID
 - The promising world of OpenID
 - Signing up for an OpenID
 - Enabling OpenID authentication with Spring Security
 - Additional required dependencies
 - Configuring OpenID support in Spring Security
 - Adding OpenID users
 - CalendarUserDetailsService lookup by OpenID
 - The OpenID user registration problem
 - How are OpenID identifiers resolved
 - Implementing user registration with OpenID
 - Registering OpenIDAuthenticationUserService
 - Attribute Exchange
 - Enabling AX in Spring Security OpenID
 - Configuring different attributes for each OpenID Provider
 - Usability enhancements
 - Automatic redirection to the OpenID Provider
 - Conditional automatic redirection
 - Is OpenID Secure
 - Summary
 - Up
- Chapter 9: Single Sign-on with Central Authentication Service
 - Introducing Central Authentication Service
 - High-level CAS authentication flow
 - Spring Security and CAS
 - Required dependencies

CAS installation and configuration
Configuring basic CAS integration
Creating the CAS ServiceProperties object
Adding the CasAuthenticationEntryPoint
Enabling CAS ticket verification
Proving authenticity with the CasAuthenticationProvider
Single logout
Configuring single logout
Clustered environments
Proxy ticket authentication for stateless services
Configuring proxy ticket authentication
Using proxy tickets
Authenticating proxy tickets
Customizing the CAS Server
CAS Maven WAR Overlay
How CAS internal authentication works
Configuring CAS to connect to our embedded LDAP server
Getting UserDetails from a CAS assertion
Returning LDAP attributes in the CAS Response
Mapping LDAP attributes to CAS attributes
Authorizing CAS Services to access custom attributes
Getting UserDetails from a CAS assertion
GrantedAuthorityFromAssertionAttributesUser Details Service
Alternative ticket authentication using SAML 1.1
How is attribute retrieval useful
Additional CAS capabilities
Summary
Up
Chapter 10: Fine-grained Access Control
Maven dependencies
Spring Expression Language (SpEL) integration
WebSecurityExpressionRoot
Using the request attribute
Using hasIpAddress
MethodSecurityExpressionRoot
Page-level authorization
Conditional rendering with Spring Security tag library
Conditional rendering based on URL access rules
Conditional rendering using SpEL
Using controller logic to conditionally render content
WebInvocationPrivilegeEvaluator
What is the best way to configure in-page authorization
Method-level security
Why we secure in layers
Securing the business tier
Adding @PreAuthorize method annotation
Instructing Spring Security to use method annotations
Validating method security
Interface-based proxies
JSR-250 compliant standardized rules
Method security using Spring's @Secured annotation
Method security rules using aspect-oriented programming
Method security rules using bean decorators
Method security rules incorporating method parameters

- Method security rules incorporating returned values
- Securing method data through role-based filtering
- Pre-filtering collections with @PreFilter
- Comparing method authorization types
- Practical considerations for annotation-based security
- Method security on Spring MVC controllers
- Class-based proxies
- Class-based proxy limitations
- Summary

Up

Chapter 11: Access Control Lists

- Using access control lists for business object security
- Access control lists in Spring Security
- Basic configuration of Spring Security ACL support
- Maven dependencies
- Defining a simple target scenario
- Adding ACL tables to the H2 database
- Configuring SecurityExpressionHandler
- AclPermissionCacheOptimizer
- PermissionEvaluator
- JdbcMutableAclService
- BasicLookupStrategy
- EhCacheBasedAclCache
- ConsoleAuditLogger
- AclAuthorizationStrategyImpl
- Creating a simple ACL entry
- Advanced ACL topics
- How permissions work
- Custom ACL permission declaration
- Enabling your JSPs with the Spring Security JSP tag library through ACL
- Mutable ACLs and authorization
- Adding ACLs to newly created Events
- Considerations for a typical ACL deployment
- About ACL scalability and performance modelling
- Do not discount custom development costs
- Should I use Spring Security ACL
- Summary

Up

Chapter 12: Custom Authorization

- How requests are authorized
- Configuration of access decision aggregation
- Configuring to use a UnanimousBased access decision manager
- Expression-based request authorization
- Customizing request authorization
- Dynamically defining access control to URLs
- JdbcRequestConfigMappingService
- FilterInvocationServiceSecurityMetadataSource
- BeanPostProcessor to extend namespace configuration
- Removing our <intercept-url> elements
- Creating a custom expression
- CustomWebSecurityExpressionRoot
- CustomWebSecurityExpressionHandler
- Configuring and using CustomWebSecurityExpressionHandler
- How does method security work

Creating a custom PermissionEvaluator
CalendarPermissionEvaluator
Configuring CalendarPermissionEvaluator
Securing our CalendarService
Benefits of a custom PermissionEvaluator
Summary

Up

Chapter 13: Session Management
Configuring session fixation protection
Understanding session fixation attacks
Preventing session fixation attacks with Spring Security
Simulating a session fixation attack
Comparing session-fixation-protection options
Restricting the number of concurrent sessions per user
Configuring concurrent session control
Understanding concurrent session control
Testing concurrent session control
Configuring expired session redirect
Common problems with concurrency control
Preventing authentication instead of forcing logout
Other benefits of concurrent session control
Displaying active sessions for a user
How Spring Security uses the HttpSession
HttpSessionSecurityContextRepository
Configuring how Spring Security uses HttpSession
Debugging with Spring Security's DebugFilter
Summary

Up

Chapter 14: Integrating with Other Frameworks
Integrating with Java Server Faces (JSF)
Customizations to support AJAX
DelegatingAuthenticationEntryPoint
AjaxRequestMatcher
Http401EntryPoint
Configuration updates
JavaScript updates
Proxy-based authorization with JSF
Custom login page in JSF
Spring Security Facelets tag library
Google Web Toolkit (GWT) integration
Spring Roo and GWT
Spring Security setup
GwtAuthenticationEntryPoint
GWT client updates
AuthRequestTransport
AuthRequiredEvent
LoginOnAuthRequired
Configuring GWT
Spring Security configuration
Method security
Method security with Spring Roo
Authorization with AspectJ
Summary

Up

Chapter 15: Migration to Spring Security 3.1
Migrating from Spring Security 2
Enhancements in Spring Security 3
Changes to configuration in Spring Security 3
Rearranged AuthenticationManager configuration
New configuration syntax for session management options
Changes to custom filter configuration
Changes to CustomAfterInvocationProvider
Minor configuration changes
Changes to packages and classes
Updates in Spring Security 3.1
Summary
Up
Appendix: Additional Reference Material
Getting started with the JBCP Calendar sample code
Creating a new workspace
Sample code structure
Importing the samples
Running the samples in Spring Tool Suite
Creating a Tomcat v7.0 server
Starting the samples within Spring Tool Suite
Shutting down the samples within Spring Tool Suite
Removing previous versions of the samples
Using HTTPS within Spring Tool Suite
Default URLs processed by Spring Security
Logical filter names migration reference
HTTPS setup in Tomcat
Generating a server certificate
Configuring Tomcat Connector to use SSL
Basic Tomcat SSL termination guide
Supplimentary materials
Up
Index
• • • • • (收起)

[Spring Security 3.1_ 下载链接1](#)

标签

Spring

Security

权限控制

LDAP

CAS

评论

只能给到3.5

Robert Winch大神的一本很有料的书，Spring Security 入门必备。

比较系统的描述了各个模块

觉得讲得一般，不过很细致就是了

[Spring Security 3.1_下载链接1](#)

书评

看了两遍，的确和前言上说的一样，是目前市场上唯一一本以Spring Security为核心的书。对Spring Security的框架介绍的比较详细。但对于现在常用的基于数据库进行角色控制部分缺乏介绍，即对于默认的基于xml的SecurityMetadataSource改造成基于DB的实现，算是一点小小的遗憾吧。

曾经在没有读此书的情况加，完全参照spring security的官方文档和网上搜索的资料搭建起了系统的authentication和authentication，至今运行稳定。之前的项目使用的是纯xml配置，在接触到spring boot后，都是用java配置，由于没有完整的了解spring security的结构以及一些基本的...

安全方面的框架比较少，前一段时间使用spring security，“不得不”研究了一下，很不幸，这是个spring名字下比较不那么好的框架，即难学又难用。如果重来，真的还不如自己实现。套用这本书里的一句话：do NOT discount custom development costs。安全无非是两个方向 - authe...

花了大概两天（10个小时）读完了这本书，期间也读了一下spring security的官方文档。基本上没有全部读到，只是抓住主要的结构读一下，理解了50%左右，可能有许多细节问题没有看到，以后在慢慢的实践中完善这方面的知识。现在只是在心中对于spring security的结构有了一些了解。

[Spring Security 3.1_下载链接1](#)