Shiro学习笔记--shiro概览

|  |  |
| --- | --- |
|  | 🢂 内容概览 |
|  | Why：此文档用来做什么？它存在的意义是什么？为解决什么问题？   |  | | --- | |  |   What：当前包含了那些内容？   |  | | --- | |  |   How：此文档应如何参考？   |  | | --- | |  |   Who：此文档适用于那些人员阅读参考？   |  | | --- | |  | |

目录

[1 Shiro概览 3](#_Toc445124608)

[1.1 什么是shiro？ 3](#_Toc445124609)

[1.2 设计目标（简单、易理解） 3](#_Toc445124610)

[1.3 功能架构 3](#_Toc445124611)

[1.4 概念模型 4](#_Toc445124612)

[1.5 系统架构 4](#_Toc445124613)

[1.6 Shiro开发包说明 6](#_Toc445124614)

# Shiro概览

## 什么是shiro？

Shiro是一个Java实现的权限管理框架，它基于RBAC模型，提供了简单易用的API，能否方便的实现：权限认证、授权、会话管理、数据加密等功能，并且提供了对web、缓存、并发的支持。

## 设计目标（简单、易理解）

1. 简单：shiro的首要目标是实现尽可能简单、易理解的权限管理功能。可以适用与任何应用，包括最简单的命令行应用和大规模集群的web应用等；
2. 与具体业务解耦：shiro仅仅实现了权限管理相关的功能框架，并不提供对具体业务数据的管理，如：用户、权限数据的存储和查找等。业务和shiro的关联，需要开发人员通过配置/定制实现。

## 功能架构

|  |
| --- |
| http://shiro.apache.org/assets/images/ShiroFeatures.pngshiro功能架构 |

1. Authentication（认证）：身份认证，可以简单的理解为”登录”，或者与此相类似的功能，比如指纹识别等；（the act of proving a user is who they say they are）
2. Authorization（授权）：权限控制，即用户可以访问系统中的哪些资源，做哪些操作；（determining 'who' has access to 'what'）；
3. Session Management（会话管理）：session类似于web中的httpsession，用来在用户使用系统过程中，存放环境数据。Shiro实现了通用的session管理功能，能够在各种应用中使用，而不是仅仅局限于web等有session支持的应用。
4. Cryptography（加密支持）：以简单的方式提供加密算法，供数据加密使用，比如：对明文密码进行加/解密，提高数据访问安全性；
5. Web支持：提供简单的API，支持web应用；
6. 缓存支持：将缓存作为shiro中的第一等公民，确保操作的快速高效；能够支持主流的第三方缓存，如：ehcache等；
7. 并发支持：支持并发应用；在一个线程中开启另一个线程，能够将权限一并传播过去；
8. 测试支持：提供简化测试的API；
9. “Run As”支持：能够指定以XX用户权限执行某项操作；
10. “Remember Me”支持：能够支持”Remember Me”功能；

## 概念模型

Shiro由Subject、SecurityManager、Realm三个基本概念组成，其关系如下图：

|  |
| --- |
| http://shiro.apache.org/assets/images/ShiroBasicArchitecture.pngshiro基本概念模型 |

1. Subject（主体）：与系统交互的用户，可以是人，也可以是第三方系统、守护进程、定时任务等；所有的Subject都绑定到一个SecurityManager对象上。当用户通过Subject的API交互时，实际的操作都是通过SecurityManager进行管理和控制的；
2. SecurityManager：安全管理器，Shiro架构中的核心控制类，用来控制和协调内部的各种组件的运行；
3. Realms（领域）：作为shiro和具体业务数据连接的纽带，可以理解为一个数据源；一个SecurityManager可以配置多个Realms，但至少需要一个；Shiro提供了一系列拆箱即用的Realms，用来适配不同类型的数据源，比如：LDAP文件系统，关系数据库，文本文件（.ini,proterties等）；如果现有realms不满足需求，也可以自己定制Realms；

## 系统架构

|  |
| --- |
| http://shiro.apache.org/assets/images/ShiroArchitecture.pngshiro系统架构 |

1. Subject ([org.apache.shiro.subject.Subject](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/subject/Subject.html))：主体，作为用户与系统交互的入口；用户与系统大部分交互都是通过Subject的API进行的；Subject由SecurityManager创建和管理；
2. SecurityManager ([org.apache.shiro.mgt.SecurityManager](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/mgt/SecurityManager.html))：安全管理器，Shiro的核心控制器，管理内部各个组件的运行；一个应用对应一个SecurityManager对象；SecurityManager被设计成低耦合的，其中的模块都针对接口编程，shiro提供各个模块的默认实现类。可以能够通过setter或者INI配置文件等方式注入具体的实现类。
3. Realms ([org.apache.shiro.realm.Realm](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/realm/Realm.html))：用来提供对安全相关数据的访问，比如登录认证和访问控制等；针对不同的数据源需要有不同的Realm实现；
4. Authenticator ([org.apache.shiro.authc.Authenticator](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/authc/Authenticator.html))：认证器，负责用户认证逻辑；能够从各种不同的Realm实现中获取用户的认证数据，并判断认证是否通过；
5. Authentication Strategy ([org.apache.shiro.authc.pam.AuthenticationStrategy](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/authc/pam/AuthenticationStrategy.html))：认证策略，当配置了多个Realm时，该类用来决定Authenticator的认证策略，即定义什么情况下认证成功，什么情况下认证失败，比如：所有realm认证成功，才算成功；或者只要有一个realm认证成功，就算成功；
6. Authorizer ([org.apache.shiro.authz.Authorizer](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/authz/Authorizer.html))：授权管理器，用来管理用户权限和角色，判断用户是否具备某个权限的访问资格；和Authenticator一样，Authorizer也可以支持多个realm配置；
7. SessionManager ([org.apache.shiro.session.mgt.SessionManager](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/session/mgt/SessionManager.html))：会话管理器，负责创建和管理用户会话生命周期；当现有应用中已经存在session环境时（如：servlet容器中的HttpSession），shiro可以使用已经存在的session；如果不存在，则能够通过创建内建的Enterprise Session Management提供独立的session环境；session中的数据能够通过SessionDao能够将session持久化到数据源；
8. SessionDAO ([org.apache.shiro.session.mgt.eis.SessionDAO](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/session/mgt/eis/SessionDAO.html))：负责将SessionManager的数据持久化到数据源，不同的数据源需要使用不同的实现类；
9. CacheManager ([org.apache.shiro.cache.CacheManager](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/cache/CacheManager.html))：缓存管理器，创建并管理缓存的生命周期，并可以将缓存提供给其它组件使用；因为认证、权限管理都需要与后台数据交互，所以shiro将缓存作为架构中的首要元素，以保证程序的性能；shiro能够支持多种第三方缓存；
10. Cryptography ([org.apache.shiro.crypto.\*](http://shiro.apache.org/static/current/apidocs/org/apache/shiro/crypto/package-summary.html))：提供了一系列易用的加密算法、哈希算法等，回避了高深的数学算法，能够被凡人所理解和使用；

## Shiro开发包说明

shiro的下载路径：<http://shiro.apache.org/download.html>

|  |  |
| --- | --- |
| Jar包 | 说明 |
| shiro-core | shiro的核心模块，提供基本的权限管理功能以及各个子模块的默认实现；  依赖slf4j-api和具体的日志实现类；  如果使用INI方式进行配置，则依赖commons-beanutils；（shiro实现了简单的注入框架） |
| shiro-web | 支持web应用； |
| shiro-aspectj | 支持aspectj |
| shiro-cas | 支持CAS |
| shiro-ehcache | 支持ehcache缓存； |
| shiro-featrues | 支持OSGi，karaf； |
| shiro-guice | 支持Google Guice框架； |
| shiro-quartz | 支持基于quartz的本地session有效性检查；（由于http无状态，系统无法知道session是否有效，定时检查session是否失效，如果失效，则清理） |
| shiro-spring | 支持spring; |
| shiro-tools-hasher-cli | 基于命令行的hash码生成器，能够对文件、流、密码等生成hash码；（可用于生成明文密码的加密字符串） |