Spring Security AppInfo Plugin - Reference Documentation

Burt Beckwith

Version 3.0.0

Table of Contents

1. Introduction to the Spring Security AppInfo Plugin 1
   1.1. Release History 1
2. Security Configuration UI 2
   2.1. Configuration 2
   2.2. Mappings 2
   2.3. Current Authentication 3
   2.4. User Cache 4
   2.5. Filter Chains 4
   2.6. Logout Handlers 5
   2.7. Voters 5
   2.8. Authentication Providers 6
3. General Notes 7
   3.1. Securing Access 7

Chapter 1. Introduction to the Spring Security AppInfo Plugin

The Spring Security AppInfo plugin provides a UI to inspect your security configuration.

If you already have the spring-security-ui plugin installed you shouldn't install this plugin, since it's part of that plugin. It's split out here into its own for users who want this information but not the entire UI plugin.

1.1. Release History

- December 8, 2015
  - 3.0.0 release
- August 16, 2015
  - 3.0.0.M1 release
- October 5, 2013
  - 2.0-RC2 release
- February 13, 2010
  - initial 1.0 release

Chapter 2. Security Configuration UI

The plugin has one controller (`SecurityInfoController.groovy`) and is available by navigating to `/securityInfo`. There are eight menus:

2.1. Configuration

The Configuration menu item displays all security-related attributes in `grails-app/conf/application.groovy`. The names omit the `grails.plugin.springsecurity` prefix:

Config | Mappings | Auth | User Cache | Filter Chains | Logout Handlers | Voters | Authentication Providers

Name	Value
active	true
adh.ajaxErrorPage	/login/ajaxDenied
adh.errorPage	/login/denied
adh.useForward	true
afterInvocationManagerProviderNames	[]
ajaxCheckClosure	
ajaxHeader	X-Requested-With
anon.key	foo
apf.allowSessionCreation	true
apf.continueChainBeforeSuccessfulAuthentication	false
apf.filterProcessesUrl	/login/authenticate
apf.passwordParameter	password
apf.postOnly	true
apf.storeLastUsername	false
apf.usernameParameter	username
atr.anonymousClass	grails. plugin. springsecurity. authentication. GrailsAnonymousAuthenticationToken

2.2. Mappings

The Mappings menu item displays the current request mapping mode (Annotation, Requestmap, or Static) and all current mappings:

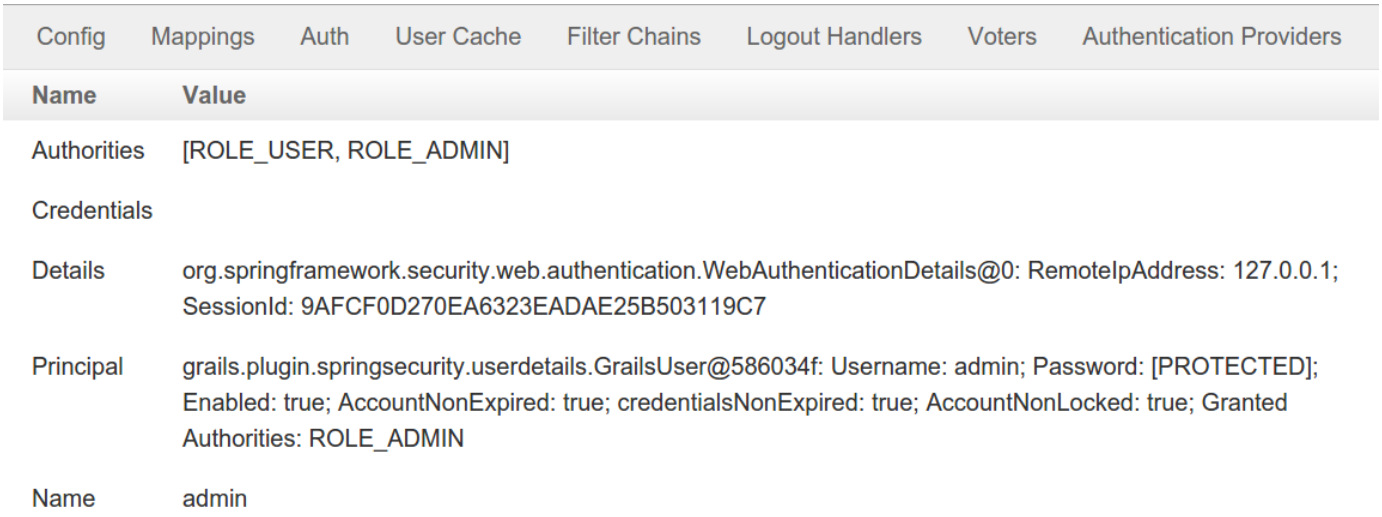
Config | Mappings | Auth | User Cache | Filter Chains | Logout Handlers | Voters | Authentication Providers

SecurityConfigType: Annotation

Pattern	ConfigAttributes	HTTP Method
/	permitAll	N/A
/error	permitAll	N/A
/index	permitAll	N/A
/index.gsp	permitAll	N/A
/shutdown	permitAll	N/A
/assets/**	permitAll	N/A
/**/js/**	permitAll	N/A
/**/css/**	permitAll	N/A
/**/images/**	permitAll	N/A
/**/favicon.ico	permitAll	N/A
/securityinfo	ROLE_ADMIN	N/A
/securityinfo/**	ROLE_ADMIN	N/A
/secure/index	ROLE_USER	N/A
/secure/index.*	ROLE_USER	N/A
/secure/index/**	ROLE_USER	N/A

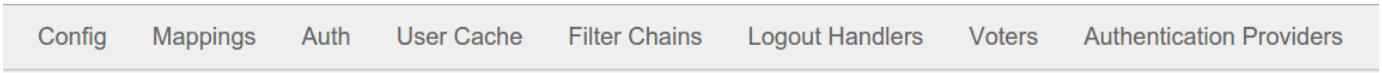
2.3. Current Authentication

The Current Authentication menu item displays your `Authentication` information, mostly for reference to see what a typical one contains:



2.4. User Cache

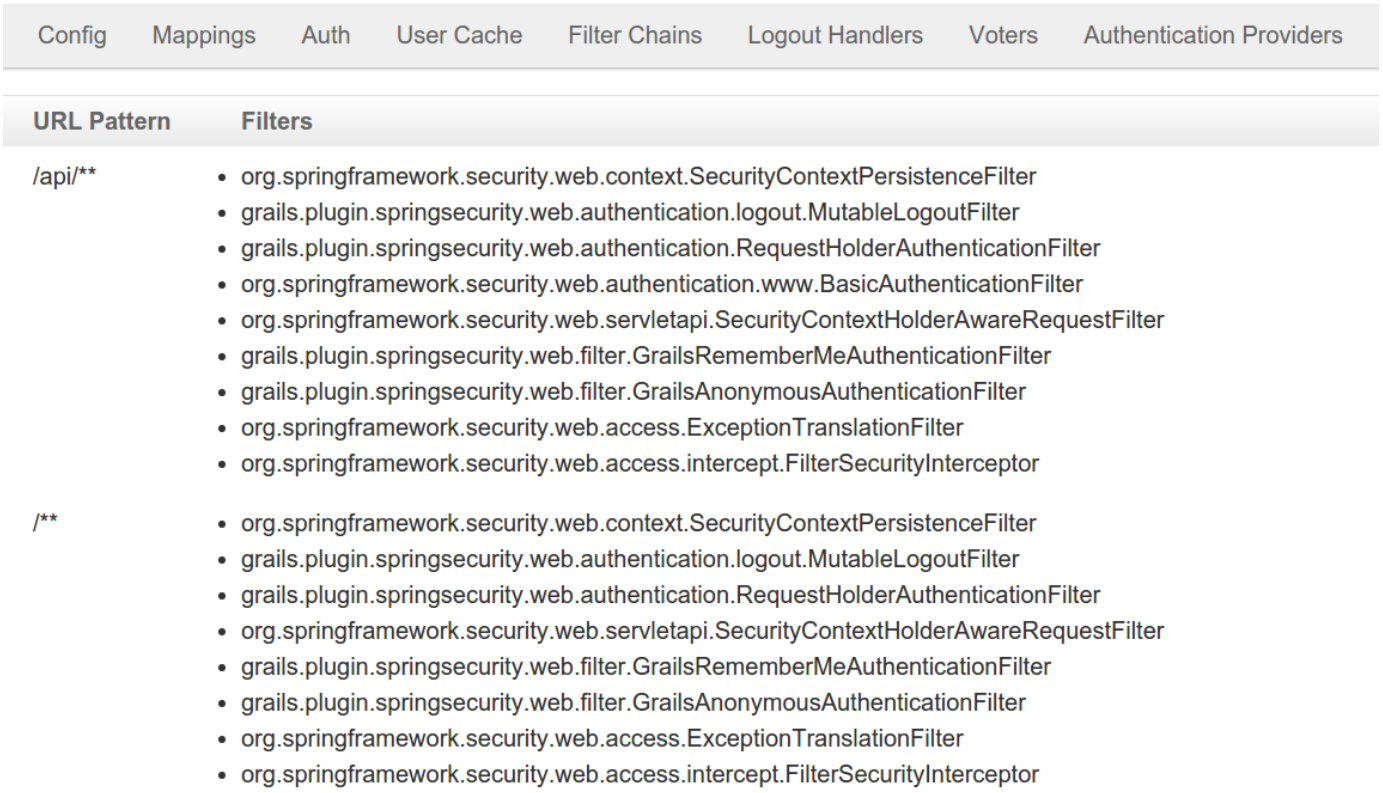
The User Cache menu item displays information about cached users (this feature is disabled by default):



Not Caching Users

2.5. Filter Chains

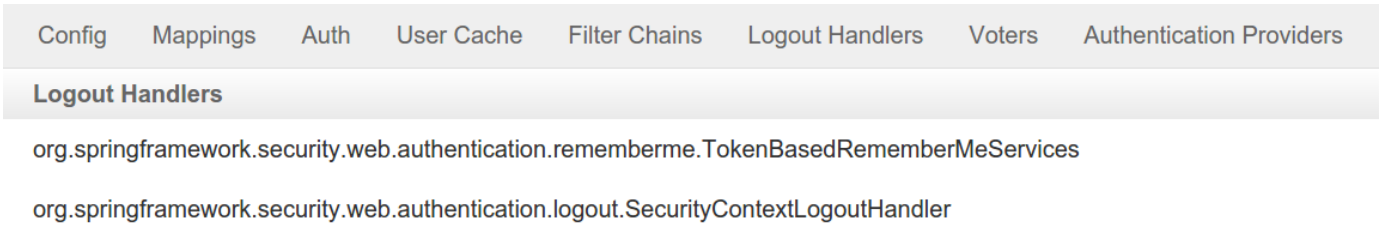
The Filter Chains menu item displays your configured Filter chains. Typically there is just one chain, applied to all URLs



It is possible to have multiple URL patterns each with its own filter chain, for example when using HTTP Basic Auth for a web service.

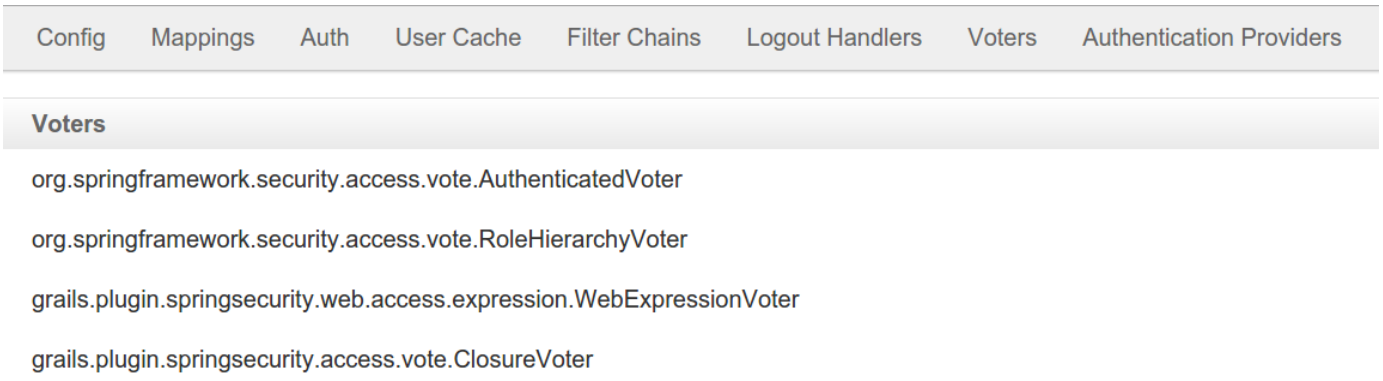
2.6. Logout Handlers

The Logout Handlers menu item displays your registered `LogoutHandler`s. Typically there will be just the two shown here, but you can register your own custom implementations, or a plugin might contribute one or more:



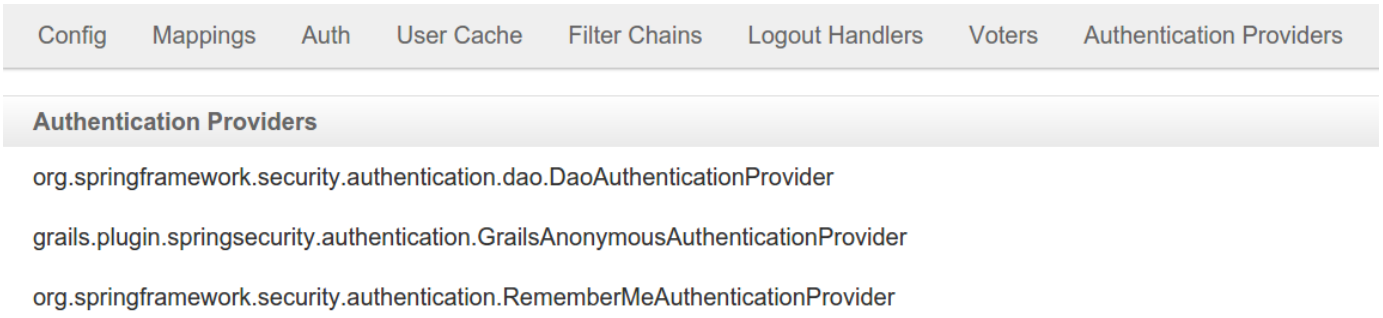
2.7. Voters

The Voters menu item displays your registered `AccessDecisionVoter`s. Typically there will be just the three shown here, but you can register your own custom implementations, or a plugin might contribute one or more:



2.8. Authentication Providers

The Authentication Providers menu item displays your registered `AuthenticationProvider`s. Typically there will be just the three shown here, but you can register your own custom implementations, or a plugin (e.g. LDAP) might contribute one or more:



Chapter 3. General Notes

3.1. Securing Access

Be sure to guard access to the `/securityInfo` url since only authorized users should have access to this information. If you're using annotations, you can register mappings in the `staticRules` property in `grails-app/conf/application.groovy`:

```
grails.plugin.springsecurity.controllerAnnotations.staticRules = [
   ...
   [pattern: '/securityinfo',    access: 'ROLE_ADMIN'],
   [pattern: '/securityinfo.*',  access: 'ROLE_ADMIN'],
   [pattern: '/securityinfo/**', access: 'ROLE_ADMIN']
]
```

If you use database Requestmaps, create new ones:

```
new Requestmap(url: '/securityinfo', configAttribute: 'ROLE_ADMIN').save()
new Requestmap(url: '/securityinfo.*', configAttribute: 'ROLE_ADMIN').save()
new Requestmap(url: '/securityinfo/**', configAttribute: 'ROLE_ADMIN').save()
```

And if you use the `interceptUrlMap` approach, add mappings to that property in `grails-app/conf/application.groovy`:

```
grails.plugin.springsecurity.interceptUrlMap = [
   ...
   [pattern: '/securityinfo',    access: 'ROLE_ADMIN'],
   [pattern: '/securityinfo.*',  access: 'ROLE_ADMIN'],
   [pattern: '/securityinfo/**', access: 'ROLE_ADMIN']
   ...
]
```