Table of Contents

The Spring Security Rest plugin fires events exactly like Spring Security Core does.

Event Notification

You can set up event notifications in two ways. The sections that follow describe each approach in more detail.

1. Register an event listener, ignoring events that do not interest you. Spring allows only partial event subscription; you use generics to register the class of events that interest you, and you are notified of that class and all subclasses.
2. Register one or more callback closures in `grails-app/conf/Config.groovy` that take advantage of the plugin's `grails.plugin.springsecurity.rest.RestSecurityEventListener`. The listener does the filtering for you.

AuthenticationEventPublisher

Spring Security REST publishes events using an AuthenticationEventPublisher which in turn fire events using the ApplicationEventPublisher. By default no events are fired since the `AuthenticationEventPublisher` instance registered is a `grails.plugin.springsecurity.rest.authentication.NullRestAuthenticationEventPublisher`. But you can enable event publishing by setting `grails.plugin.springsecurity.useSecurityEventListener = true` in `grails-app/conf/Config.groovy`.

You can use the `useSecurityEventListener` setting to temporarily disable and enable the callbacks, or enable them per-environment.

Token Creation

Currently the Spring Security REST plugin supports a single event in addition to the default spring security events. The event is fired whenever a new token is created. See `grails.plugin.springsecurity.rest.RestTokenCreationEvent`

Every time a token is successfully submitted, an `AuthenticationSuccessEvent` will be fired.

Registering an Event Listener

Enable events with `grails.plugin.springsecurity.useSecurityEventListener = true` and create one or more Groovy or Java classes, for example:

Listing 1. Custom event listener

```
package com.foo.bar

import org.springframework.context.ApplicationListener
import grails.plugin.springsecurity.rest.RestTokenCreationEvent

class MySecurityEventListener
      implements ApplicationListener<RestTokenCreationEvent> {

   void onApplicationEvent(RestTokenCreationEvent event) {
      // The access token is a delegate of the event, so you have
access to an instance of
`grails.plugin.springsecurity.rest.token.AccessToken`
   }
}
```

Register the class in `grails-app/conf/spring/resources.groovy`:

Listing 2. Event listener registration

```
import com.foo.bar.MySecurityEventListener

beans = {
   mySecurityEventListener(MySecurityEventListener)
}
```

Registering Callback Closures

Alternatively, enable events with `grails.plugin.springsecurity.useSecurityEventListener = true` and register one or more callback closure(s) in `grails-app/conf/Config.groovy` and let `SecurityEventListener` do the filtering.

Implement the event handlers that you need, for example:

Listing 3. Callback closures

```
grails.plugin.springsecurity.useSecurityEventListener = true

grails.plugin.springsecurity.onRestTokenCreationEvent = { e, appCtx
->
   // handle RestTokenCreationEvent
}
```

None of these closures are required; if none are configured, nothing will be called. Just implement the event handlers that you need.