What's New in spring-security-core 2.0

Burt Beckwith






I'VE MADE A
HUGE MISTAKE
generator.net

What's new to me?

What's new to me?

Two pages of commits by Lari and Graeme, plus some PRs

First, what's old

Grails-y wrapper around Spring Security

First, what's old

Many defaults, lots of configurability – designed to be customized and extended

First, what's old

Easy to get started – add dependency in BuildConfig.groovy and run `s2-quickstart`

First, what's old

Helper classes
(`SpringSecurityService`,
taglibs, controllers, etc.)

First, what's old

Form, HTTP Basic, Digest auth

First, what's old

Users, roles, hierarchical roles, customizable `UserDetailsService`

First, what's old

Many password hashing options, including options for salted passwords

First, what's old

Remember-me

First, what's old

- Ajax support

First, what's old

Switch-user (similar to “sudo”)

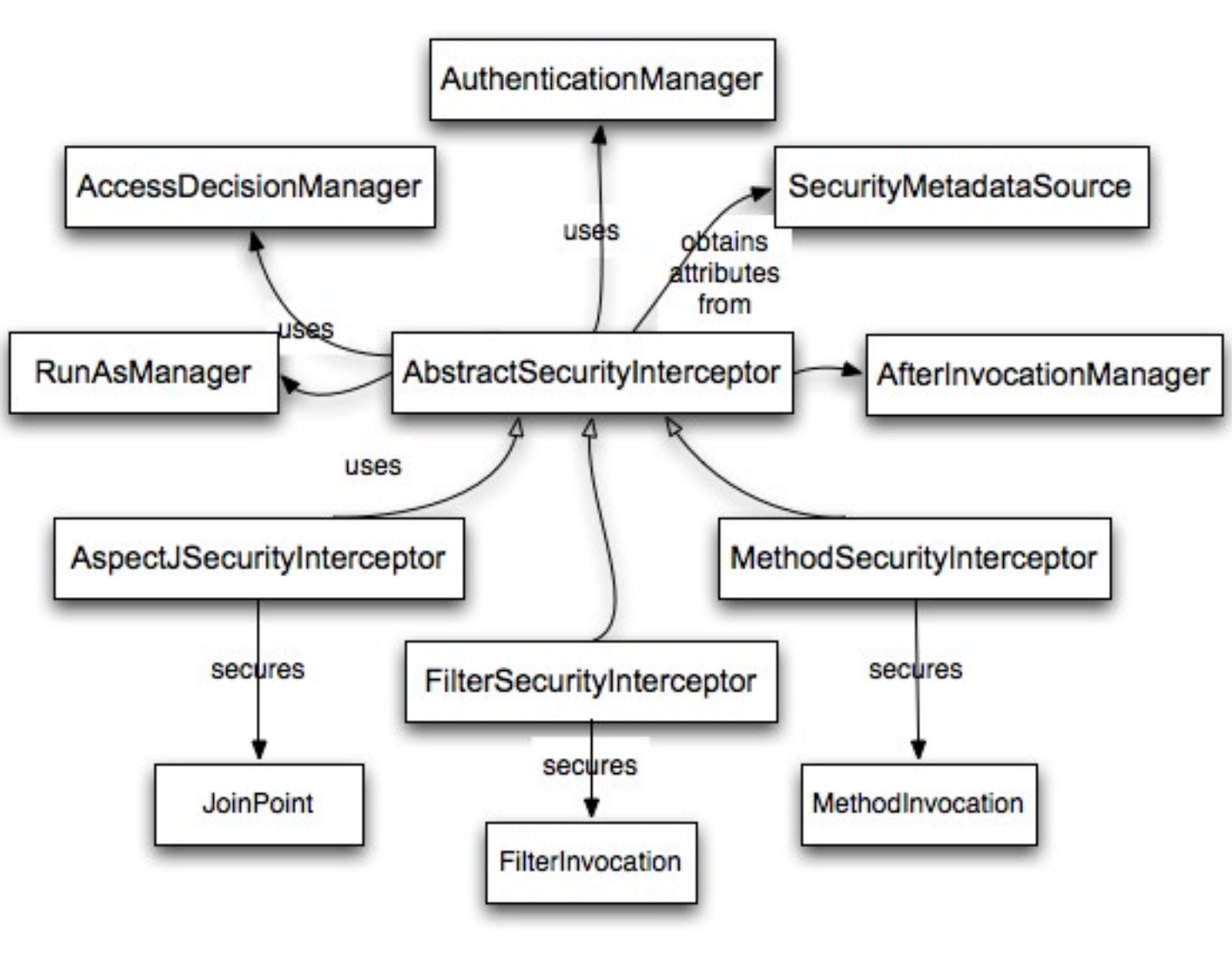
First, what's old

HTTP/HTTPS channel security

First, what's old

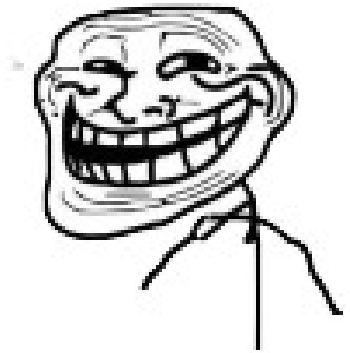
Session Fixation Protection

First, what's old



- Convention over configuration, with centralized configuration in grails-app/conf/Config.groovy
- Highly configurable and customizable
- Registers Spring Security beans in application context, filters in web.xml
- Storing users, roles, and optionally requestmaps in the database, with access through domain classes
- Guarding URLs with annotations, requestmap domain class, or static configuration
- Password encryption (with support for salt)
- "Remember me" cookie
- Security tags; <g:ifAllGranted/>, <g:ifNotGranted/>, <g:ifLoggedIn/>, etc.
- Security service; encodePassword(), isLoggedIn(), etc.
- Multiple authentication providers
    - Form-based
    - HTTP Basic
    - Browser certificate (x509)
- Switch User
- Channel security
- IP address restrictions
- Ajax login
- Convenient event handlers
- Digest authentication
- Session Fixation Prevention
- Salted passwords
- Hierarchical roles
- Account locking and forcing password change
- Mostly Java for performance

- Convention over configuration, with centralized configuration in grails-app/conf/Config.groovy
- Highly configurable and customizable
- Registers Spring Security beans in application context, filters in web.xml
- Storing users, roles, and optionally requestmaps in the database, with access through domain classes
- Guarding URLs with annotations, requestmap domain class, or static configuration
- Password encryption (with support for salt)
- "Remember me" cookie
- Security tags; <g:ifAllGranted/>, <g:ifNotGranted/>, <g:ifLoggedIn/>, etc.
- Security service; encodePassword(), isLoggedIn(), etc.
- Multiple authentication providers
  - Form-based
  - HTTP Basic
  - Browser certificate (x509)
- Switch User
- Channel security
- IP address restrictions
- Ajax login
- Convenient event handlers
- Digest authentication
- Session Fixation Prevention
- Salted passwords
- Hierarchical roles
- Account locking and forcing password change
- Mostly Java for performance



Trust me, it has a lot of features

First, what's old

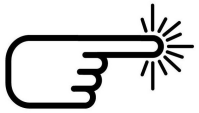
Extension plugins (ACL, CAS, LDAP, OpenID, UI, etc.)

First, what's old

And more!

So … what's new?

See the notes in the docs:



What's New in Version 2.0

Highlights

More aggressively secure by default

Highlights

More aggressively secure by default

Pessimistic Lockdown by default, use `grails.plugin.springsecurity.rejectIfNoRule` and `grails.plugin.springsecurity.fii.rejectPublicInvocations` to configure

Highlights

Pessimistic Lockdown:

```
grails.plugin.springsecurity.
controllerAnnotations.staticRules = [
   '/':                ['permitAll'],
   '/index':           ['permitAll'],
   '/index.gsp':       ['permitAll'],
   '/**/js/**':        ['permitAll'],
   '/**/css/**':       ['permitAll'],
   '/**/images/**':    ['permitAll'],
   '/**/favicon.ico':  ['permitAll']
]
```

Highlights

Pessimistic Lockdown:

```
for (String url in [
      '/', '/index', '/index.gsp',
      '/**/favicon.ico', '/**/js/**',
      '/**/css/**', '/**/images/**',
      '/login', '/login.*', '/login/*',
      '/logout', '/logout.*',
      '/logout/*']) {
   new Requestmap(
     url: url,
     configAttribute: 'permitAll')
    .save()
}
```

Highlights

Pessimistic Lockdown:

```
grails.plugin.springsecurity.
interceptUrlMap = [
    '/':                    ['permitAll'],
    '/index':               ['permitAll'],
    '/index.gsp':           ['permitAll'],
    '/**/js/**':            ['permitAll'],
    '/**/css/**':           ['permitAll'],
    '/**/images/**':        ['permitAll'],
    '/**/favicon.ico':      ['permitAll'],
    '/login/**':            ['permitAll'],
    '/logout/**':           ['permitAll']
]
```

Highlights

More aggressively secure by default

Logout uses POST only, configure with

`grails.plugin.springsecurity.logout.postOnly`

Highlights

More aggressively secure by default

Default password hash is now **bcrypt**, and **PBKDF2** is also available (`password.algorithm = 'pbkdf2'`)

Highlights

More aggressively secure by default

Session Fixation Prevention is enabled by default, configure with `grails.plugin.springsecurity.useSessionFixationPrevention`

Highlights

Using Spring Security 3.2.3.RELEASE - originally 3.1, then 3.2.0-RC1, now 3.2.3 as of this week

Highlights

Package changes

Highlights

Package changes

Everything now under
`grails.plugin.springsecurity`

Highlights

Package changes

Subpackages are similar to Spring Security packages

Highlights

Package changes

Subpackages are similar to Spring Security packages

e.g. `GormUserDetailsService` → `grails.plugin.springsecurity.userdetails.GormUserDetailsService`

Highlights

Package changes

Subpackages are similar to Spring Security packages

e.g. `AjaxAwareAccessDeniedHandler` → `grails.plugin.springsecurity.web.access.AjaxAwareAccessDeniedHandler`

Highlights

Configuration prefix changes

`grails.plugins.springsecurity` → `grails.plugin.springsecurity`

Highlights

No HQL (except in UI plugin), all queries use “where” and Criteria

Highlights

More configurable properties in Spring beans (goal is ~100%)

Highlights

More `private` → `protected`

Highlights

SpringSecurityService updates:

No `withTransaction`, using `@Transactional` as needed

Highlights

SpringSecurityService updates:

`getCurrentUser()` uses `get(principal.id)` if principal is **`GrailsUser`**, otherwise `findWhere((usernamePropName): principal.username)`

Highlights

SpringSecurityService updates:

New `loadCurrentUser()` method

```
class SomeController {

   def springSecurityService

   def someAction() {
      def user = springSecurityService.isLoggedIn() ?
          springSecurityService.loadCurrentUser() : null
      if (user) {
         CreditCard card = CreditCard.findByIdAndUser(
               params.id as Long, user)
         ...
      }
      ...
   }
}
```

NoStackUsernameNotFoundException

```
package grails.plugin.springsecurity.userdetails;

import org.springframework.security.core.userdetails.
UsernameNotFoundException;

public class NoStackUsernameNotFoundException
       extends UsernameNotFoundException {

   private static final long serialVersionUID = 1;

   public NoStackUsernameNotFoundException() {
      super("User not found");
   }

   @Override
   public synchronized Throwable fillInStackTrace() {
      // do nothing
      return this;
   }
}
```

New `@Authorities` annotation

Helps make your annotations more DRY - see http://burtbeckwith.com/blog/?p=1398

Guarding URLs

Guarding URLs

`@Secured` now only works with controller methods

Guarding URLs

`@Secured` supports Closures

```
@Secured(closure = {
    assert request
    assert ctx
    authentication.name == 'admin1'
})
def someMethod() {
    …
}
```

Guarding URLs

All 3 approaches support HTTP verbs

```
@Secured(
  value=["hasRole('ROLE_ADMIN')"],
  httpMethod='POST')
def someMethod() {
    …
}
```

Anonymous Authentication

Principal now is a `UserDetails` like when you're authenticated, but with `ROLE_ANONYMOUS`

I18N

User-contributed Russian, Norwegian Bokmål, Brazilian Portuguese (pt-BR), Italian, and Swedish translations

Controllers and GSPs

LoginController.groovy, LogoutController.groovy, auth.gsp, denied.gsp are in the plugin now - copy to app to customize

Support for Grails 2.3

Support for redirect mappings, and `@Secured` in `RestfulController`

Support for Grails 2.4

Removed a use of `ApplicationHolder` (using `Holders` instead)

New `DebugFilter`

Based on `org.springframework.security.config.debug.DebugFilter` - enable with `debug.useFilter` (only in dev!)

Role Groups

```
grails s2-quickstart
com.yourapp User Role
--groupClassName=RoleGroup
```

Role Groups

Adds to Config.groovy:

```
grails.plugin.springsecurity.
authority.groupAuthorityNameField =
'authorities'

grails.plugin.springsecurity.
useRoleGroups = true
```

Role Groups

Adds 3 new domain classes:

- `RoleGroup`
- `RoleGroupRole` (`RoleGroup` <-> `Role` many-many join class)
- `UserRoleGroup` (`RoleGroup` <-> `User` many-many join class)

Role Groups

Changes `User.getAuthorities()`

```
Set<Role> getAuthorities() {
   UserRole.findAllByUser(this)
      .collect { it.role }
}
```

→

```
Set<RoleGroup> getAuthorities() {
   UserRoleGroup.findAllByUser(this)
      .collect { it.roleGroup }
}
```

Role Groups

New docs:

- Group Class
- PersonGroup Class
- GroupAuthority Class

Miscellaneous

Using bcrypt impl from Spring Security instead of copied code

Miscellaneous

`GrantedAuthorityImpl` is deprecated, use `SimpleGrantedAuthority`

Miscellaneous

```
provided ':webxml:1.4.1' →
compile ':webxml:1.4.1'
```

Miscellaneous

Grails 2.0+ only

Miscellaneous

Generated User class **`enabled`** property now defaults to **`true`**

```
def u = new User(
    username: 'me',
    password: 'itsasecret')
   .save()
```

Miscellaneous

Only prints status messages (e.g. "Configuring Spring Security Core ...") if `printStatusMessages` is not false

Miscellaneous

No default values for `userLookup.userDomainClassName`, `authority.className`, etc. - error messages now make more sense

Miscellaneous

`AuthenticationDetailsSource`

details class isn't configurable in Spring Security 3.2, so the docs describe how to customize

Miscellaneous

You can configure the `SecurityContextHolder` strategy (defaults to `ThreadLocal`, but can use `InheritableThreadLocal` or a custom impl - configure with `sch.strategyName`

Miscellaneous

Spring Security 3.2 doesn't store the last username in the HTTP session – to use the old behavior configure with `apf.storeLastUsername`

Miscellaneous

Functional tests now use Geb

Miscellaneous

The 1.x code is in its own branch

New Config Properties

- printStatusMessages = true
- ajaxCheckClosure = null
- afterInvocationManagerProviderNames = []
- authority.groupAuthorityNameField = null
- useRoleGroups = false
- apf.storeLastUsername = false
- logout.clearAuthentication = true
- logout.invalidateHttpSession = true
- logout.targetUrlParameter = null
- logout.alwaysUseDefaultTargetUrl = false
- logout.redirectToReferer = false
- logout.postOnly = true

New Config Properties

- failureHandler.allowSessionCreation = true
- successHandler.useReferer = false
- adh.useForward = true
- password.hash.iterations = 10000
- rememberMe.createSessionOnSuccess = true
- requestMap.httpMethodField = 'httpMethod'
- basic.credentialsCharset = 'UTF-8'
- switchUser.usernameParameter = SwitchUserFilter.SPRING_SECURITY_SWITCH_USERNAME_KEY
- x509.subjectDnClosure = null
- debug.useFilter = false
- sch.strategyName = SecurityContextHolder.MODE_THREADLOCAL

New Config Properties

- scr.allowSessionCreation = true
- scr.disableUrlRewriting = true
- scr.springSecurityContextKey = HttpSessionSecurityContextRepository.SPRING_SECURITY_CONTEXT_KEY
- scpf.forceEagerSessionCreation = false
- fii.alwaysReauthenticate = false
- fii.rejectPublicInvocations = true
- fii.validateConfigAttributes = true
- fii.publishAuthorizationSuccess = false
- fii.observeOncePerRequest = true

Changed Config Properties

- rejectIfNoRule → true
- userLookup.userDomainClassName → null
- userLookup.authorityJoinClassName → null
- useSessionFixationPrevention → true
- password.algorithm 'SHA-256' → 'bcrypt'
- rememberMe.persistentToken.domainClassName → null
- rememberMe.useSecureCookie false → null
- requestMap.className → null
- atr.anonymousClass → GrailsAnonymousAuthenticationToken
- providerManager.eraseCredentialsAfterAuthentication → true

Removed Config Properties

- requestCache.onlyOnGet
- authenticationDetails.authClass
- anon.userAttribute
- controllerAnnotations.matcher
- controllerAnnotations.lowercase
- filterChain.stripQueryStringFromUrls

So, what's left to do?

So, what's left to do?

A lot. 31 issues scheduled for 2.0

So, what's left to do?

A lot. 31 issues scheduled for 2.0

But many are simple, and there will probably be an RC3 release

¡Gracias!

http://cuteoverload.files.wordpress.com/2014/03/cute-smiling-animals-251.jpg