Table of contents

Spring Security Kerberos plugin

Spring Security Kerberos Plugin - Reference Documentation

Authors: Burt Beckwith

Version: 1.0-RC1

Table of Contents

1 Introduction to the Spring Security Kerberos Plugin

1.1 History

2 Usage

3 Configuration

1 Introduction to the Spring Security Kerberos Plugin

The Kerberos plugin adds Kerberos single sign-on support to a Grails application that uses Spring Security. It depends on the Spring Security Core plugin.

Once you have configured a Kerberos server (typically Microsoft Active Directory or MIT Kerberos) and have configured your Grails application(s) as clients, users who are have authenticated at the Kerberos server will be automatically authenticated as a user of your application(s) without requiring a password.

1.1 History

- Version 1.0-RC1
  - released October 24, 2013
- Version 0.1
  - released January 30, 2011

2 Usage

> ⚠ Configuring your Kerberos server is beyond the scope of this document. There are several options and this will most likely be done by IT staff. It's assumed here that you already have a running Kerberos server.

The plugin uses the Kerberos/SPNEGO Spring Security extension and the most relevant information about it can be found in this blog post.

There isn't much that you need to do in your application to be a Kerberos client. Just install this plugin, and configure the two required parameters and whatever optional parameters you want in `Config.groovy`. These are described in detail in guide:3. Configuration but typically you only need to set these properties

```
grails.plugin.springsecurity.kerberos.ticketValidator.servicePrincipal =
      'HTTP/kerberos.server.name@KERBEROS.DOMAIN'

grails.plugin.springsecurity.kerberos.ticketValidator.keyTabLocation =
      'file:///path/to/your.keytab'
```

UserDetailsService

Currently the only information that is retrieved from Kerberos is the username (plus the authentication status of course) so you'll need to have user and role data in your database corresponding to Kerberos users. Since you'll be authenticating externally you can either remove the password field from the user class and use a custom `UserDetailsService` or just store dummy values in the password column to satisfy the not-null constraint.

3 Configuration

There are a few configuration options for the Kerberos plugin.

The plugin uses the Kerberos/SPNEGO Spring Security extension and the most relevant information about it can be found in this blog post.

All of these property overrides must be specified in `grails-app/conf/Config.groovy` using the `grails.plugin.springsecurity` suffix, for example

```
grails.plugin.springsecurity.kerberos.debug = true
```

There are two required properties:

Name	Default	Meaning
kerberos.ticketValidator.servicePrincipal	none, required	the web application service principal, e.g. `HTTP/www.example.com@EXAMPLE.COM`
kerberos.ticketValidator.keyTabLocation	none, required	the URL to the location of the keytab file containing the service principal's credentials, e.g. `file:///etc/http-web.keytab`

and three optional properties:

Name	Default	Meaning
kerberos.configLocation	`null`	The location of the Kerberos config file (specify the path to the file, but omit "file://", e.g. "c:/krb5.conf"). Leave unset to use the default location (e.g. `/etc/krb5.conf`, `c:winntkrb5.ini`, `/etc/krb5/krb5.conf`)
kerberos.debug	`false`	if `true` enables debug logs from the Sun Kerberos Implementation
kerberos.ticketValidator.debug	`false`	if `true` enables ticket validator debug messages