Spring Security Kerberos Plugin - Reference Documentation

Authors: Burt Beckwith

Version: 1.0.0

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 <u>Kerberos</u> single sign-on support to a Grails application that uses Spring Securidepends on the <u>Spring Security Core plugin</u>.

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

1.1 History

- Version 1.0.0
 - released December 7, 2015
- 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 adds support for Kerberos and is based on the **Spring Security Kerberos extension**.

There isn't much that you need to do in your application to be a Kerberos client. Just install this plugin configure the two required parameters and whatever optional parameters you want in Config.gro 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'
```

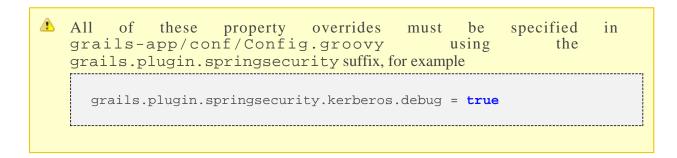
UserDetailsService

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

3 Configuration

There are a few configuration options for the Kerberos plugin.

The plugin uses the **Spring Security Kerberos extension**.



There are two required properties:

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

and some optional properties:

Name	Default	Meaning
kerberos.active	true	set to false to disable the plugin
kerberos.client.debug	false	if true enables debug logs for kerberos client bean
kerberos.configLocation	null	The location of the Kerberos co file (specify the path to the file, omit "file://", e.g. "c:/krb5.co Leave unset to use the default loca (e.g. /etc/krb5.conf, c:winntkrb5.ini, /etc/krb5/krb5.conf)
kerberos.debug	false	if true enables debug logs for kerberosConfig bean
kerberos.skipIfAlreadyAuthenticated	true	if true s SpnegoAuthenticationProcessingF processing if already authenticated
kerberos.spnegoEntryPointForwardUrl	null	if set (e.g. '/login/auth') the EntryF will forward there in addition setting the WWW-Authenti header
kerberos.successHandler.headerName	'WWW-Authenticate'	the name of the header to following successful authenticatio
kerberos.successHandler.headerPrefix	'Negotiate '	the prefix for the encoded respetoken value
kerberos.ticketValidator.debug	false	if true enables debug logs for ticketValidator bean
kerberos.ticketValidator.holdOnToGSSContext	false	if true, hold on to the GSS section context, otherwise call dispose immediately