Installation et configuration du serveur CAS (archive)

Installer un JDK, Maven et Tomcat comme spécifié sur cette page : Installation Java, Maven et Tomcat

Installation d'un serveur CAS basique

Installation

Télécharger la dernière version de CAS depuis http://www.jasig.org/cas/download et décompresser :

```
[root@cas ~]# cd /usr/local
[root@cas local]# wget http://www.ja-sig.org/downloads/cas/cas-server-3.3.5-release.tar.gz
--2010-01-18 10:47:55--  http://www.ja-sig.org/downloads/cas/cas-server-3.3.5-release.tar.gz
Resolving www.ja-sig.org... 128.112.131.108
Connecting to www.ja-sig.org|128.112.131.108|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 14467126 (14M) [application/x-gzip]
Saving to: "cas-server-3.3.5-release.tar.gz"
100%[======================================================================>] 14,467,126   271K/s   in 84s
2010-01-18 10:49:19 (168 KB/s) - "cas-server-3.3.5-release.tar.gz" saved [14467126/14467126]
[root@cas local]# tar xf cas-server-3.3.5-release.tar.gz
[root@cas local]# cd cas-server-3.3.5
[root@cas cas-server-3.3.5]# cd cas-server-webapp
[root@cas cas-server-webapp]#
```

Modifier le fichier **src/main/webapp/WEB-INF/classes/log4j.properties** en indiquant le chemin des logs :

```
log4j.appender.logfile.File=/var/log/tomcat5/cas.log
```

Générer le WAR, le copier dans Tomcat et redémarrer :

```
[root@cas cas-server-webapp]#  mvn package install
[root@cas cas-server-webapp]#  cp target/cas.war /var/lib/tomcat5/webapps/ROOT.war
[root@cas cas-server-webapp]#  /etc/init.d/tomcat5 restart
```

Debuggage

Ajouter dans le fichier **src/main/webapp/WEB-INF/classes/log4j.properties** la ligne suivante :

```
log4j.logger.org.jasig.cas=DEBUG
```

Les logs se trouvent dans le répertoire **/var/log/tomcat5**.

La mise au point la plus difficile est celle de Kerberos. Les logs en debug de **Krb5LoginModule** se trouvent dans **catalina.out**.

Test

Désactiver si nécessaire le firewall pour le port 8080 (**system-config-firewall**) et tester http://cas.ifsic.univ-rennes1.fr:8080 (user = test, password = test).

Passage en HTTPS

Génération du keystore

Rapatrier *Jetty* (par exemple dans **/usr/local**) depuis http://static.roopindersingh.com/jetty-6.1.7.jar .

Copier les clés publique (**cas.ifsic.univ-rennes1.fr.pem**) et privée (**cas.ifsic.univ-rennes1.fr.key**) dans **/etc/pki/tls/private** pour générer le keystore dans **/etc/tomcat5** (en donnant comme mot de passe changeit) :

```
[root@cas private]# openssl pkcs12 -export \
>     -out cas.ifsic.univ-rennes1.fr.pkcs12 \
>     -in cas.ifsic.univ-rennes1.fr.pem \
>     -inkey cas.ifsic.univ-rennes1.fr.key
Enter Export Password:
Verifying - Enter Export Password:
[root@cas private]# java -cp /usr/local/jetty-6.1.7.jar org.mortbay.jetty.security.PKCS12Import \
>     cas.ifsic.univ-rennes1.fr.pkcs12 /etc/tomcat5/cas.ifsic.univ-rennes1.fr.keystore
Enter input keystore passphrase: changeit
Enter output keystore passphrase: changeit
Alias 0: 1
Adding key for alias 1
[root@cas private]#
```

Régler les permissions du keystore :

```
[root@cas private]# cd /etc/tomcat5/
[root@cas tomcat5]# chgrp tomcat cas.ifsic.univ-rennes1.fr.keystore
[root@cas tomcat5]# chmod 640 cas.ifsic.univ-rennes1.fr.keystore
[root@cas tomcat5]#
```

Configuration de Tomcat

Dans **/etc/tomcat5/server.xml**, commenter le connecteur HTTP sur le port 8080 et décommenter le connecteur HTTPS sur le port 8443 en ajoutant l'attribut :

```
keystoreFile="/etc/tomcat5/cas.ifsic.univ-rennes1.fr.keystore"
```

Test

Redémarrer Tomcat, désactiver si nécessaire le firewall pour le port 8443 et tester https://cas.ifsic.univ-rennes1.fr:8443 (user = test, password = test).

Ajout de l'authentification LDAP

Script de déploiement de CAS

Pour faciliter le déploiement du serveur CAS, on pourra ajouter le script **/usr/local/cas-server-3.3.5/deploy.sh** suivant :

```
#!/bin/bash
/etc/init.d/tomcat5 stop
pushd /usr/local/cas-server-3.3.5/cas-server-webapp
mvn package install && rm -f /var/lib/tomcat5/webapps/ROOT.war && cp target/cas.war /var/lib/tomcat5/webapps
/ROOT.war
popd
/etc/init.d/tomcat5 start
```

Configuration de CAS pour LDAP

Dans le fichier **src/main/webapp/WEB-INF/deployerConfigContext.xml**, ajouter le bean suivant pour déclarer le contexte LDAP :

```
<bean id="contextSource" class="org.springframework.ldap.core.support.LdapContextSource">
  <property name="pooled" value="true"/>
  <property name="urls">
    <list>
      <value>ldap://ldapglobal.univ-rennes1.fr/</value>
    </list>
  </property>
  <property name="userDn" value=""/>
  <property name="password" value=""/>
  <property name="baseEnvironmentProperties">
    <map>
      <entry>
       <key>
         <value>java.naming.security.authentication</value>
       </key>
       <value>simple</value>
      </entry>
     </map>
  </property>
</bean>
```

puis changer le handler **SimpleTestUsernamePasswordAuthenticationHandler** par celui-ci :

```
<bean
   class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
     <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
     <property name="contextSource" ref="contextSource" />
</bean>
```

Test

Redéployer le serveur CAS et tester l'authentification d'un utilisateur LDAP.

Ajout d'un frontal Apache

On va dans cette partie configurer un frontal Apache sur le port 443, qui va accéder au Tomcat du serveur CAS en AJP sur le port 8009.

Il n'est pas obligatoire de mettre un frontal Apache devant Tomcat, mais cela délègue le chiffrement à Apache au lieu de Tomcat et simplifie l'admisnitration système (cette architecture est employée de manière générale sur les plateformes d'exploitation).

Configuration de Apache

Installer le certificat du serveur en éditant **/etc/httpd/conf.d/ssl.conf** et modifier les lignes suivantes dans le *virtual host* **_default_:443** :

```
#SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateFile /etc/pki/tls/private/cas.ifsic.univ-rennes1.fr.pem
#SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
SSLCertificateKeyFile /etc/pki/tls/private/cas.ifsic.univ-rennes1.fr.key
```

Ajouter la ligne suivante pour indiquer à Apache de passer les requêtes à Tomcat :

```
ProxyPass / ajp://cas.ifsic.univ-rennes1.fr:8009/ min=0 max=100 smax=50 ttl=10 route=ori-indexing
```

Configuration de Tomcat

S'assurer que le connecteur AJP sur le port 8009 est décommenté et a bien le paramètres **tomcatAuthentication** positionné à **false** :

```
<Connector port="8009"
    debug="0"
    enableLookups="false"
    redirectPort="8443"
    protocol="AJP/1.3"
    tomcatAuthentication="false" />
```

Le connecteur HTTPS sur le port 8443 peut être commenté.

Test

Le serveur CAS doit désormais répondre sur l'URL https://cas.ifsic.univ-rennes1.fr (sur le port 443 par défaut en HTTPS).

Ajout de l'authentification Kerberos

La documentation de référence est http://www.ja-sig.org/wiki/display/CASUM/SPNEGO .

Configuration de Kerberos

Procéder comme vu précédemment pour générer le fichier /etc/http.keytab qui sera utilisé ultérieurement par la librairie JCIFS :

```
[root@cas ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  ktadd -k /etc/http.keytab HTTP/cas.ifsic.univ-rennes1.fr
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS mode with 96-bit
SHA-1 HMAC added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS mode with 96-bit
SHA-1 HMAC added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc mode with HMAC
/sha1 added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with HMAC/md5 added to
keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1 added to
keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with RSA-MD5 added
to keytab WRFILE:/etc/http.keytab.
kadmin:  exit
[root@cas ~]
```

Configuration de CAS

Ajouter le support du handler spnego

Editer le fichier **<cas-home>/cas-server-webapp/pom.xml** et ajouter la dépendance suivante (par exemple juste après la dépendance vers le module **cas-server-support-ldap**) :

```
<dependency>
     <groupId>${project.groupId}</groupId>
     <artifactId>cas-server-support-spnego</artifactId>
     <version>${project.version}</version>
</dependency>
```

Modifier le login webflow

Editer le fichier **<cas-home>/cas-server-webapp/src/main/webapp/WEB-INF/login-webflow.xml** et ajouter l'état suivant juste avant l'état *viewLoginForm* :

```
<action-state id="startAuthenticate">
  <action bean="negociateSpnego" />
  <transition on="success" to="spnego" />
</action-state>

<action-state id="spnego">
  <action bean="spnego" />
  <transition on="success" to="sendTicketGrantingTicket" />
  <transition on="error" to="viewLoginForm" />
</action-state>
```

Dans ce même fichier, remplacer les références à *viewLoginForm* par *remoteAuthenticate* pour les trois decision-state *gatewayRequestCheck* et *renewRequestCheck* :

```
<decision-state id="gatewayRequestCheck">
  <if
    test="${externalContext.requestParameterMap['gateway'] != '' &amp;&amp; externalContext.requestParameterMap
['gateway'] != null &amp;&amp; flowScope.service != null}"
    then="redirect"
    else="startAuthenticate" />
</decision-state>
```

```
<decision-state id="renewRequestCheck">
  <if
    test="${externalContext.requestParameterMap['renew'] != '' &amp;&amp; externalContext.requestParameterMap
['renew'] != null}"
    then="startAuthenticate"
    else="generateServiceTicket" />
</decision-state>
```

Déclarer le bean implémentant le nouvel état du webflow en ajoutant les lignes suivantes dans le fichier **<cas-home>/cas-server-webapp/src/main/webapp/WEB-INF/cas-servlet.xml** (par exemple juste avant le bean **authenticationViaFormAction**) :

```
<bean
    id="negociateSpnego"
    class="org.jasig.cas.support.spnego.web.flow.SpnegoNegociateCredentialsAction" />

<bean
    id="spnego"
    class="org.jasig.cas.support.spnego.web.flow.SpnegoCredentialsAction">
        <property name="centralAuthenticationService" ref="centralAuthenticationService"/>
</bean>
```

Modifier le schéma d'authentification

Pour modifier le schéma d'authentification, éditer le fichier **<cas-home>/cas-server-webapp/src/main/webapp/WEB-INF/deployerConfigContext.xml** et modifier le bean **authenticationManager** en ajoutant :

- **PrincipalBearingCredentialsToPrincipalResolver** après les *resolvers* existants de **credentialsToPrincipalResolvers**
- **PrincipalBearingCredentialsAuthenticationHandler** avant les *handlers* existants de **authenticationHandlers**

```
<bean id="authenticationManager" class="org.jasig.cas.authentication.AuthenticationManagerImpl">
  <property name="credentialsToPrincipalResolvers">
    <list>
      <!--  ... the others credentialsToPrincipalResolvers ... -->
      <bean class="org.jasig.cas.support.spnego.authentication.principal.SpnegoCredentialsToPrincipalResolver"
/>
    </list>
  </property>
  <property name="authenticationHandlers">
    <list>
      <bean class="org.jasig.cas.support.spnego.authentication.handler.support.JCIFSSpnegoAuthenticationHandler"
>
        <property name="authentication">
          <bean class="jcifs.spnego.Authentication" />
        </property>
        <property name="principalWithDomainName" value="true" />
        <property name="NTLMallowed" value="false"/>
      </bean>
      <!--  ... the others authenticationHandlers... -->
    </list>
  </property>
</bean>
```

Le bean **authenticationManager** doit ainsi ressembler à :

```
<bean   id="authenticationManager"
        class="org.jasig.cas.authentication.AuthenticationManagerImpl">
    <property name="credentialsToPrincipalResolvers">
      <list>
        <bean class="org.jasig.cas.authentication.principal.UsernamePasswordCredentialsToPrincipalResolver" />
        <bean class="org.jasig.cas.authentication.principal.HttpBasedServiceCredentialsToPrincipalResolver" />
        <bean class="org.jasig.cas.support.spnego.authentication.principal.
SpnegoCredentialsToPrincipalResolver" />
      </list>
    </property>
    <property name="authenticationHandlers">
      <list>
        <bean class="org.jasig.cas.support.spnego.authentication.handler.support.
JCIFSSpnegoAuthenticationHandler">
          <property name="authentication">
            <bean class="jcifs.spnego.Authentication" />
          </property>
          <property name="principalWithDomainName" value="true" />
          <property name="NTLMallowed" value="false"/>
      </bean>
      <bean class="org.jasig.cas.authentication.handler.support.
HttpBasedServiceCredentialsAuthenticationHandler"
              p:httpClient-ref="httpClient" />
        <bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
          <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
          <property name="contextSource" ref="contextSource" />
        </bean>
      </list>
    </property>
</bean>
```

Ajouter enfin le bean **jcifsConfig**, qui done les options de configuration de JCIFS :

```
<bean name="jcifsConfig" class="org.jasig.cas.support.spnego.authentication.handler.support.JCIFSConfig">
  <property
      name="jcifsServicePrincipal"
      value="HTTP/cas.ifsic.univ-rennes1.fr" />
  <property
      name="kerberosDebug"
      value="true" />
  <property
      name="kerberosRealm"
      value="UNIV-RENNES1.FR" />
  <property
      name="kerberosKdc"
      value="kerb.ifsic.univ-rennes1.fr" />
  <property
      name="loginConf"
      value="/usr/local/cas-server-3.3.5/cas-server-webapp/src/main/webapp/WEB-INF/login.conf" />
```

Configuration de JCIFS

La configuration de JCIFS se fait également dans le fichier login.conf pointé par le bean **jcifsConfig**. Créer ce fichier avec le contenu suivant :

```
jcifs.spnego.initiate {
  com.sun.security.auth.module.Krb5LoginModule
  required
  useKeyTab=true
  keyTab="/etc/http.keytab"
};
jcifs.spnego.accept {
  com.sun.security.auth.module.Krb5LoginModule
  required
  useKeyTab=true
  keyTab="/etc/http.keytab"
};
```

On peut également ajouter l'option **debug=true** pour obtenir des informations dans **catalina.out**.

Configuration de Tomcat

Il faut passer à la JVM qui exécute Tomcat l'option **-Djavax.security.auth.useSubjectCredsOnly=false**, par exemple en éditant le fichier **/etc/tomcat5/tomcat5.conf** et en ajoutant la ligne suivante :

```
JAVA_OPTS="$JAVA_OPTS -Djavax.security.auth.useSubjectCredsOnly=false"
```

Test

Un navigateur bien configuré et possédant des credentials Kerberos valides doit maintenant se connecter au serveur CAS sans aucune interaction....