链滴

CAS4.0+LDAP 配置说明

作者：cbxjj

原文链接：https://ld246.com/article/1493284998454

来源网站：链滴

前置说明

CAS的配置见前文, 单点登录--CAS简单搭建, cas4.0搭建好之后，默认试用已经写死的用户名登录，即asuser/Mellon,下方将讲述cas4.0和ldap的整合

添加相关jar

```
<dependency>
   <groupId>org.springframework.ldap</groupId>
   <artifactId>spring-ldap-core</artifactId>
   <version>2.3.1.RELEASE</version>
</dependency>
<dependency>
   <groupId>org.jasig.cas</groupId>
   <artifactId>cas-server-support-ldap</artifactId>
   <version>4.0.0</version>
</dependency>
```

注意：cas-server-support-ldap4.0.0中默认带的ldaptive.jar的版本号为1.0.3

此版本有bug，最好换成1.0.5,修改方式

直接修改maven库目录 repository\org\jasig\cas\cas-server\4.0.0\cas-server-4.0.0.pom

```
<ldaptive.version>1.0.5</ldaptive.version>
```

修改默认的认证方式

```
<!-- <entry key-ref="primaryAuthenticationHandler" value-ref="primaryPrincipalResolver"/>
->
<entry key-ref="ldapAuthHandler" value-ref="primaryPrincipalResolver" />

<bean id="ldapAuthHandler" class="org.jasig.cas.authentication.LdapAuthenticationHandler"
      p:principalIdAttribute="cn" c:authenticator-ref="authenticator">
   <property name="principalAttributeMap">
      <map>
         <entry key="member" value="member" />
         <entry key="mail" value="mail" />
         <entry key="cn" value="cn" />
      </map>
   </property>
   <property name="allowMultiplePrincipalAttributeValues" value="true"></property>
</bean>
```

修改默认的formatResover为SearchDnResolver,已适应多ou登录

```
<bean id="authenticator" class="org.ldaptive.auth.Authenticator"
      c:resolver-ref="multiDnResolver" c:handler-ref="authHandler" />


<!--<bean id="adDnResolver" class="org.ldaptive.auth.FormatDnResolver"
    c:format="cn=%s,ou=wuhan,dc=infozr,dc=com" /> -->
```

原文链接：CAS4.0+LDAP 配置说明

```
<bean id="multiDnResolver" class="org.ldaptive.auth.SearchDnResolver"
      p:baseDn="dc=infozr,dc=com"
      p:subtreeSearch="true"
      p:allowMultipleDns="false"
      p:connectionFactory-ref="connectionFactory"
      p:userFilter="(cn={user})" />
```

完整的配置文件

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Licensed to Jasig under one or more contributor license agreements.
   See the NOTICE file distributed with this work for additional information
   regarding copyright ownership. Jasig licenses this file to you under the
   Apache License, Version 2.0 (the "License"); you may not use this file except
   in compliance with the License. You may obtain a copy of the License at the
   following location: http://www.apache.org/licenses/LICENSE-2.0 Unless required
   by applicable law or agreed to in writing, software distributed under the
   License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS
   OF ANY KIND, either express or implied. See the License for the specific
   language governing permissions and limitations under the License. -->
<!-- | deployerConfigContext.xml centralizes into one file some of the declarative
   configuration that | all CAS deployers will need to modify. | | This file
   declares some of the Spring-managed JavaBeans that make up a CAS deployment.
   | The beans declared in this file are instantiated at context initialization
   time by the Spring | ContextLoaderListener declared in web.xml. It finds
   this file because this | file is among those declared in the context parameter
   "contextConfigLocation". | | By far the most common change you will need
   to make in this file is to change the last bean | declaration to replace
   the default authentication handler with | one implementing your approach
   for authenticating usernames and passwords. + -->

<beans xmlns="http://www.springframework.org/schema/beans"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springfr
mework.org/schema/p"
   xmlns:c="http://www.springframework.org/schema/c" xmlns:tx="http://www.springframew
rk.org/schema/tx"
   xmlns:util="http://www.springframework.org/schema/util" xmlns:sec="http://www.springfr
mework.org/schema/security"
   xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springfr
mework.org/schema/beans/spring-beans-3.2.xsd
      http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx
spring-tx-3.2.xsd
      http://www.springframework.org/schema/security http://www.springframework.org/sch
ma/security/spring-security-3.2.xsd
      http://www.springframework.org/schema/util http://www.springframework.org/schema/u
il/spring-util.xsd">

   <!-- | The authentication manager defines security policy for authentication
      by specifying at a minimum | the authentication handlers that will be used
      to authenticate credential. While the AuthenticationManager | interface supports
      plugging in another implementation, the default PolicyBasedAuthenticationManager
      should | be sufficient in most cases. + -->
```

```
    <bean id="authenticationManager"
        class="org.jasig.cas.authentication.PolicyBasedAuthenticationManager">
        <constructor-arg>
            <map>
                <!-- | IMPORTANT | Every handler requires a unique name. | If more than
                    one instance of the same handler class is configured, you must explicitly
                    | set its name to something other than its default name (typically the simple
                    class name). -->
                <entry key-ref="proxyAuthenticationHandler" value-ref="proxyPrincipalResolver" />

                <!-- <entry key-ref="primaryAuthenticationHandler" value-ref="primaryPrincipalRe
olver"
                    /> -->
                <entry key-ref="ldapAuthHandler" value-ref="primaryPrincipalResolver" />
            </map>
        </constructor-arg>

        <!-- Uncomment the metadata populator to allow clearpass to capture and
            cache the password This switch effectively will turn on clearpass. <property
            name="authenticationMetaDataPopulators"> <util:list> <bean class="org.jasig.cas.ex
ension.clearpass.CacheCredentialsMetaDataPopulator"
            c:credentialCache-ref="encryptedMap" /> </util:list> </property> -->

        <!-- | Defines the security policy around authentication. Some alternative
            policies that ship with CAS: | | * NotPreventedAuthenticationPolicy - all
            credential must either pass or fail authentication | * AllAuthenticationPolicy
            - all presented credential must be authenticated successfully | * RequiredHandlerAuth
ntticationPolicy
            - specifies a handler that must authenticate its credential to pass -->
        <property name="authenticationPolicy">
            <bean class="org.jasig.cas.authentication.AnyAuthenticationPolicy" />
        </property>
    </bean>

    <!-- Required for proxy ticket mechanism. -->
    <bean id="proxyAuthenticationHandler"
        class="org.jasig.cas.authentication.handler.support.HttpBasedServiceCredentialsAuthenti
ationHandler"
        p:httpClient-ref="httpClient" />

    <!-- | TODO: Replace this component with one suitable for your enviroment.
        | | This component provides authentication for the kind of credential used
        in your environment. In most cases | credential is a username/password pair
        that lives in a system of record like an LDAP directory. | The most common
        authentication handler beans: | | * org.jasig.cas.authentication.LdapAuthenticationHandle

        | * org.jasig.cas.adaptors.jdbc.QueryDatabaseAuthenticationHandler | * org.jasig.cas.adap
ors.x509.authentication.handler.support.X509CredentialsAuthenticationHandler
        | * org.jasig.cas.support.spnego.authentication.handler.support.JCIFSSpnegoAuthenticati
nHandler -->
    <bean id="primaryAuthenticationHandler"
        class="org.jasig.cas.authentication.AcceptUsersAuthenticationHandler">
        <property name="users">
            <map>
```

```
            <entry key="casuser" value="Mellon" />
          </map>
        </property>
    </bean>

    <!-- Required for proxy ticket mechanism -->
    <bean id="proxyPrincipalResolver"
      class="org.jasig.cas.authentication.principal.BasicPrincipalResolver" />

    <!-- | Resolves a principal from a credential using an attribute repository
      that is configured to resolve | against a deployer-specific store (e.g. LDAP). -->
    <bean id="primaryPrincipalResolver"
      class="org.jasig.cas.authentication.principal.PersonDirectoryPrincipalResolver">
      <property name="attributeRepository" ref="attributeRepository" />
    </bean>

    <!-- Bean that defines the attributes that a service may return. This example
      uses the Stub/Mock version. A real implementation may go against a database
      or LDAP server. The id should remain "attributeRepository" though. + -->
    <bean id="attributeRepository"
      class="org.jasig.services.persondir.support.StubPersonAttributeDao"
      p:backingMap-ref="attrRepoBackingMap" />

    <util:map id="attrRepoBackingMap">
      <entry key="cn" value="cn" />
      <entry key="eduPersonAffiliation" value="eduPersonAffiliation" />
      <entry key="groupMembership" value="groupMembership" />
    </util:map>

    <!-- Sample, in-memory data store for the ServiceRegistry. A real implementation
      would probably want to replace this with the JPA-backed ServiceRegistry DAO
      The name of this bean should remain "serviceRegistryDao". + -->
    <bean id="serviceRegistryDao" class="org.jasig.cas.services.InMemoryServiceRegistryDaoI
pl"
      p:registeredServices-ref="registeredServicesList" />

    <util:list id="registeredServicesList">
      <bean class="org.jasig.cas.services.RegexRegisteredService" p:id="0"
        p:name="HTTP and IMAP" p:description="Allows HTTP(S) and IMAP(S) protocols"
        p:serviceId="^(https?|imaps?)://.*" p:evaluationOrder="10000001" />
      <!-- Use the following definition instead of the above to further restrict
        access to services within your domain (including sub domains). Note that
        example.com must be replaced with the domain you wish to permit. This example
        also demonstrates the configuration of an attribute filter that only allows
        for attributes whose length is 3. -->
      <!-- <bean class="org.jasig.cas.services.RegexRegisteredService"> <property
        name="id" value="1" /> <property name="name" value="HTTP and IMAP on example
com"
        /> <property name="description" value="Allows HTTP(S) and IMAP(S) protocols
        on example.com" /> <property name="serviceId" value="^(https?|imaps?)://([A-Za-z0
9_-]+\.)*example\.com/.*"
        /> <property name="evaluationOrder" value="0" /> <property name="attributeFilter
>
        <bean class="org.jasig.cas.services.support.RegisteredServiceRegexAttributeFilter"
```

```
        c:regex="^\w{3}$" /> </property> </bean> -->
    </util:list>

    <bean id="auditTrailManager"
      class="com.github.inspektr.audit.support.Slf4jLoggingAuditTrailManager" />

    <bean id="healthCheckMonitor" class="org.jasig.cas.monitor.HealthCheckMonitor"
      p:monitors-ref="monitorsList" />

    <bean id="ldapAuthHandler" class="org.jasig.cas.authentication.LdapAuthenticationHandl
r"
      p:principalIdAttribute="cn" c:authenticator-ref="authenticator">
      <property name="principalAttributeMap">
        <map>
          <!-- | This map provides a simple attribute resolution mechanism. | Keys
            are LDAP attribute names, values are CAS attribute names. | Use this facility
            instead of a PrincipalResolver if LDAP is | the only attribute source. -->
          <entry key="member" value="member" />
          <entry key="mail" value="mail" />
          <entry key="cn" value="cn" />
        </map>
      </property>
      <property name="allowMultiplePrincipalAttributeValues" value="true"></property>
    </bean>


    <bean id="authenticator" class="org.ldaptive.auth.Authenticator"
      c:resolver-ref="multiDnResolver" c:handler-ref="authHandler" />


    <bean id="adDnResolver" class="org.ldaptive.auth.FormatDnResolver"
       c:format="cn=%s,ou=wuhan,dc=infozr,dc=com" />

    <bean id="multiDnResolver" class="org.ldaptive.auth.SearchDnResolver"
        p:baseDn="dc=infozr,dc=com"
        p:subtreeSearch="true"
        p:allowMultipleDns="false"
        p:connectionFactory-ref="connectionFactory"
        p:userFilter="(cn={user})" />

    <bean id="authHandler" class="org.ldaptive.auth.PooledBindAuthenticationHandler"
      p:connectionFactory-ref="pooledLdapConnectionFactory" />


    <bean id="pooledLdapConnectionFactory" class="org.ldaptive.pool.PooledConnectionFact
ry"
      p:connectionPool-ref="connectionPool" />


    <bean id="connectionPool" class="org.ldaptive.pool.BlockingConnectionPool"
      init-method="initialize" p:poolConfig-ref="ldapPoolConfig"
      p:blockWaitTime="3000" p:validator-ref="searchValidator"
      p:pruneStrategy-ref="pruneStrategy" p:connectionFactory-ref="connectionFactory" />
```

```
    <bean id="ldapPoolConfig" class="org.ldaptive.pool.PoolConfig"
      p:minPoolSize="3" p:maxPoolSize="10" p:validateOnCheckOut="false"
      p:validatePeriodically="true" p:validatePeriod="300" />


    <bean id="connectionFactory" class="org.ldaptive.DefaultConnectionFactory"
      p:connectionConfig-ref="connectionConfig" />


    <bean id="connectionConfig" class="org.ldaptive.ConnectionConfig"
      p:ldapUrl="ldap://192.168.0.80:389 ldap://192.168.0.112:389" p:connectTimeout="3000"
      p:useStartTLS="false" /><!--p:sslConfig-ref="sslConfig" 上面内容根据自己的LDAP内容来
置 -->


    <!-- <bean id="sslConfig" class="org.ldaptive.ssl.SslConfig">
      <property name="credentialConfig">
        <bean class="org.ldaptive.ssl.X509CredentialConfig"
          p:trustCertificates="/home/keys/infozr.keystore" />  证书地址
      </property>
    </bean> -->


    <bean id="pruneStrategy" class="org.ldaptive.pool.IdlePruneStrategy"
      p:prunePeriod="300" p:idleTime="600" />


    <bean id="searchValidator" class="org.ldaptive.pool.SearchValidator" />

    <util:list id="monitorsList">
      <bean class="org.jasig.cas.monitor.MemoryMonitor"
        p:freeMemoryWarnThreshold="10" />
      <!-- NOTE The following ticket registries support SessionMonitor: * DefaultTicketRegistry

        * JpaTicketRegistry Remove this monitor if you use an unsupported registry. -->
      <bean class="org.jasig.cas.monitor.SessionMonitor"
        p:ticketRegistry-ref="ticketRegistry"
        p:serviceTicketCountWarnThreshold="5000" p:sessionCountWarnThreshold="100000"
/>
    </util:list>

</beans>


</br>
</br>
```
