# 一、配置mysql

## 1、创建数据库

# 创建测试数据库

CREATE DATABASE cas;

USE cas;

# 创建测试表

CREATE TABLE `cas\_t\_user` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`loginr\_name` varchar(50) DEFAULT NULL,

`password` varchar(255) DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=MyISAM DEFAULT CHARSET=utf8;

# 插入两条测试用户数据

INSERT INTO `cas\_t\_user` VALUES (1,'admin','96e79218965eb72c92a549dd5a330112');

INSERT INTO `cas\_t\_user` VALUES (2,'user','96e79218965eb72c92a549dd5a330112');

**其中用户的密码为：111111**

## 2、配置cas-server

修改/WEB\_INF/deployerConfigContext.xml

### 2.1、配置dataSource

// 添加以下bean 数据库用户名密码按自己的配置

<!-- 添加数据源 -->

<bean id="dataSource"

class="org.springframework.jdbc.datasource.DriverManagerDataSource">

<property name="driverClassName" value="com.mysql.jdbc.Driver" />

<property name="url" value="jdbc:mysql://localhost/cas" />

<property name="username" value="root" />

<property name="password" value=" root " />

</bean>

### 2.2、配置passwordEncoder

// 配置cas自带的MD5加密器

<!-- 配置passwordEncoder -->

<bean id="MD5PasswordEncoder"

class="org.jasig.cas.authentication.handler.DefaultPasswordEncoder">

<constructor-arg index="0" value="MD5" />

</bean>

### 2.3、配置认证类

<!-- 配置认证类 -->

<bean

id="dbAuthHandler" class="org.jasig.cas.adaptors.jdbc.QueryDatabaseAuthenticationHandler">

<!--dataSource指向上面配置的dataSource bean-->

<property name="dataSource" ref="dataSource" ></property>

<property name="sql" value="select password from sso\_t\_user where login\_name=?" ></property>

<!--passwordEncoder 指向上面配置的 passwordEncoder bean-->

<property name="passwordEncoder" ref="MD5PasswordEncoder" ></property>

</bean>

### 2.4、修改authenticationManager

// 1. 找到以下bean

<bean

id="authenticationManager" class="org.jasig.cas.authentication.PolicyBasedAuthenticationManager">

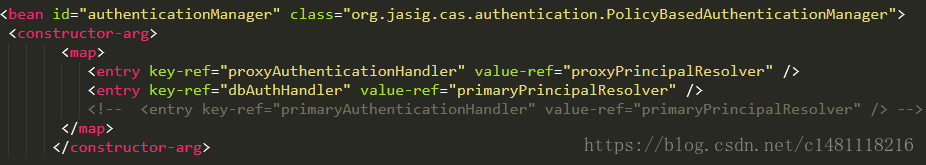
// 2. 将其中下面这个entry注释

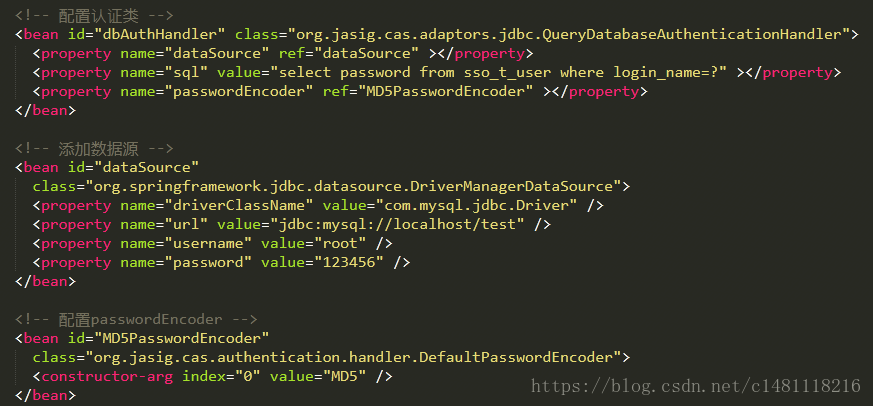
<entry key-ref="primaryAuthenticationHandler" value-ref="primaryPrincipalResolver" />

// 3. 添加下面这个entry

<entry key-ref="dbAuthHandler" value-ref="primaryPrincipalResolver" />

**修改的配置图**





### 2.5、添加相关jar包

<**dependency**>  
 <**groupId**>org.jasig.cas</**groupId**>  
 <**artifactId**>cas-server-support-jdbc</**artifactId**>  
 <**version**>${project.version}</**version**>  
 <**type**>jar</**type**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>commons-dbcp</**groupId**>  
 <**artifactId**>commons-dbcp</**artifactId**>  
 <**version**>1.4</**version**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>mysql</**groupId**>  
 <**artifactId**>mysql-connector-java</**artifactId**>  
 <**version**>5.1.6</**version**>  
</**dependency**>

## 3、启动测试

启动之后进入http://localhost:8080/cas/login

以用户： admin / 111111 登录成功 则配置成功完

---------------------

# 二、cas server去掉https验证

因为cas4.2以上的代码做了一些框架的优化，4.2以下的版本的很多配置都是写在xml文件中。4.2以上统一提取到了cas.properties文件中。

所以在去掉https的时候 方式也有所变化。

现在网上查到的资料很多都还停留在对4.2以下版本的配置修改中，我们这里把两种情况都记录下来。

## 1、4.2以上cas server去掉https验证

属性设置

我们在官网上看到有关于https的安全认证的属性有:

<https://apereo.github.io/cas/4.2.x/installation/Configuring-SSO-Session-Cookie.html>

# The encryption secret key. By default, must be a octet string of size 256.

# tgc.encryption.key=

# The signing secret key. By default, must be a octet string of size 512.

# tgc.signing.key=

# Decides whether SSO cookie should be created only under secure connections.

# tgc.secure=true

# The expiration value of the SSO cookie

# tgc.maxAge=-1

# The name of the SSO cookie

# tgc.name=TGC

# The path to which the SSO cookie will be scoped

# tgc.path=/cas

# Decides whether SSO Warning cookie should be created only under secure connections.

# warn.cookie.secure=true

# The expiration value of the SSO Warning cookie

# warn.cookie.maxAge=-1



Decides whether SSO cookie should be created only under secure connections

tgc.secure=true

Decides whether SSO Warning cookie should be created only under secure connections

warn.cookie.secure=true

决定是否需要在安全连接(https)的情况下才能创建tgc和warn.cookie。

我们这里只要把tgc.secure和warn.cookie.secure这两个属性修改成false即可。

serviceId修改

还有一个地方需要修改,否则会报Application Not Authorized to Use CAS。

src/main/resources/services中的  HTTPSandIMAPS-10000001.json

{

  "@class" : "org.jasig.cas.services.RegexRegisteredService",

  "serviceId" : "^(https|imaps)://.\*",

  "name" : "HTTPS and IMAPS",

  "id" : 10000001,

  "description" : "This service definition authorized all application urls that support HTTPS and IMAPS protocols.",

  "proxyPolicy" : {

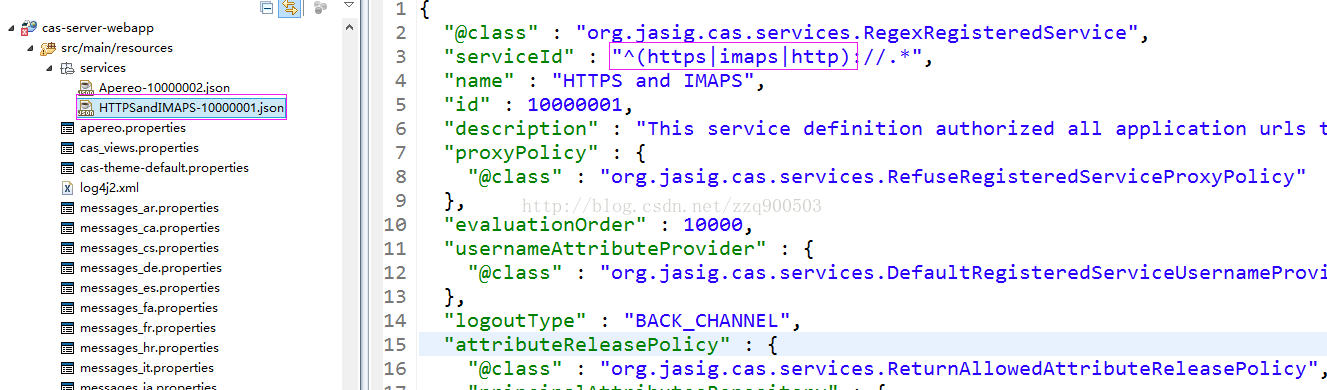
    "@class" : "org.jasig.cas.services.RefuseRegisteredServiceProxyPolicy"

  }

 "serviceId" : "^(https|imaps)://.\*"

需要修改成 "serviceId" : "^(https|imaps|http)://.\*"   或者把https修改成http。

---------------------



## 2、4.2以下cas server去掉https验证

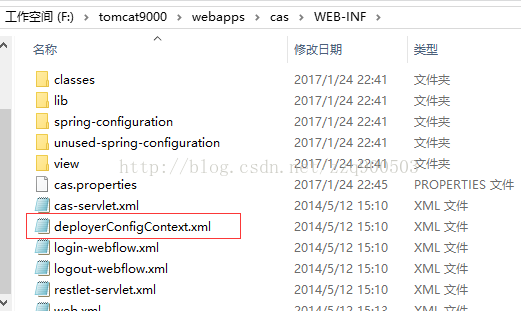
cas默认是采用https模式的，我们没有配置证书，所以要么配置证书，要么取消https的过滤，让http协议也能访问。

我们这里取消https的配置，让http也能访问。

需要修改三个地方的配置(针对4.0.0版本，其他版本的话第一处必改，其他的看看还有没有带有cookie的文件名)

**修改一deployerConfigContext.xml增加参数p:requireSecure="false"**

我们在tomcat的webapp中找到 cas/WEB-INF/deployerConfigContext.xml



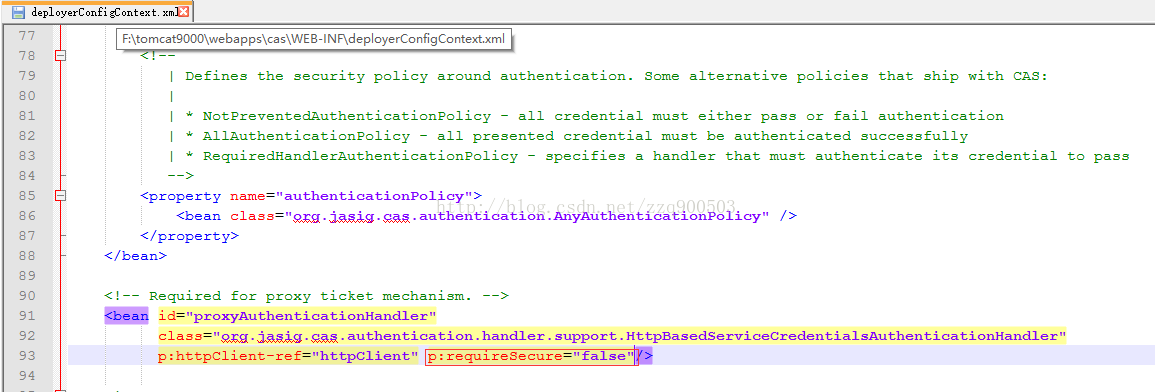
<bean class="org.jasig.cas.authentication.handler.support.HttpBasedServiceCredentialsAuthenticationHandler"

p:httpClient-ref="httpClient"/>

这里需要增加参数p:requireSecure="false"，requireSecure属性意思为是否需要安全验证，即HTTPS，false为不采用。修改后为：

  <bean class="org.jasig.cas.authentication.handler.support.HttpBasedServiceCredentialsAuthenticationHandler"

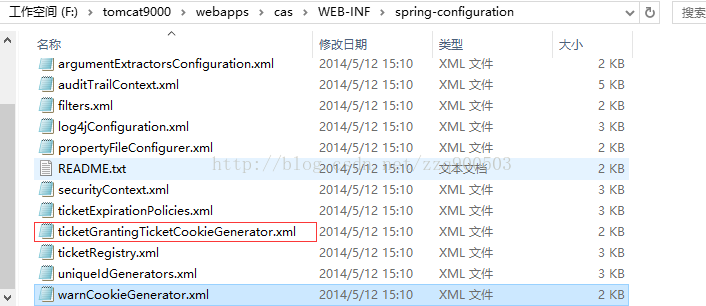
 p:httpClient-ref="httpClient"  p:requireSecure="false"/>



**修改二ticketGrantingTicketCookieGenerator.xml修改p:cookieSecure="false"**

 我们在tomcat的webapp中找到

cas/WEB-INF/spring-configuration/ticketGrantingTicketCookieGenerator.xml



<bean

id="ticketGrantingTicketCookieGenerator" class="org.jasig.cas.web.support.CookieRetrievingCookieGenerator"

      p:cookieSecure="true"

      p:cookieMaxAge="-1"

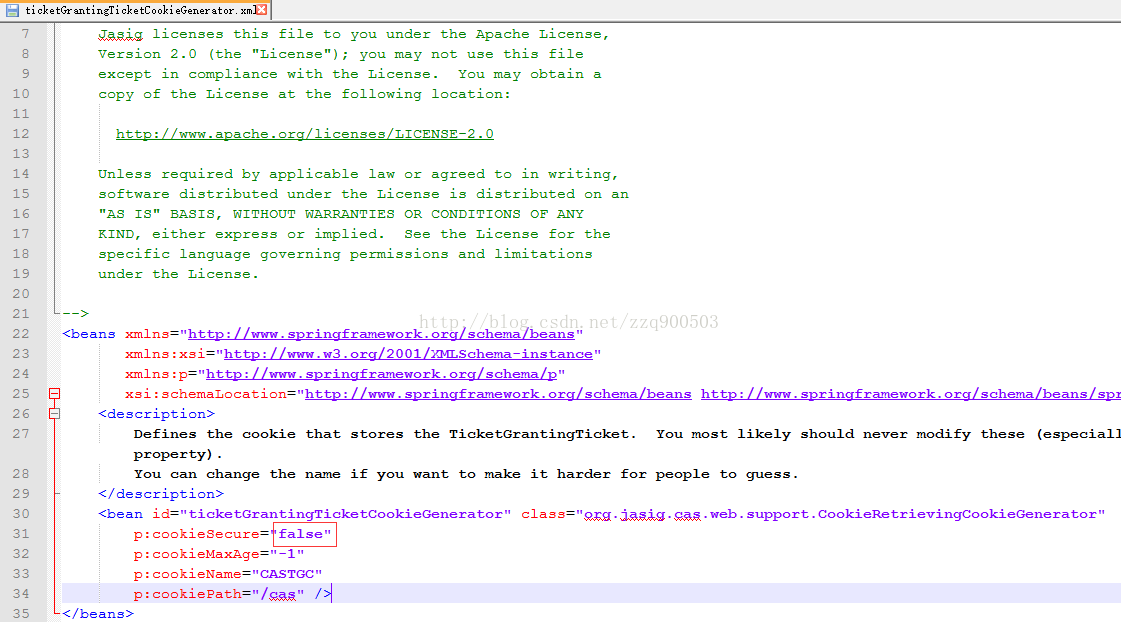
      p:cookieName="CASTGC"

      p:cookiePath="/cas" />

 参数p:cookieSecure="true"，同理为HTTPS验证相关，TRUE为采用HTTPS验证，FALSE为不采用https验证。

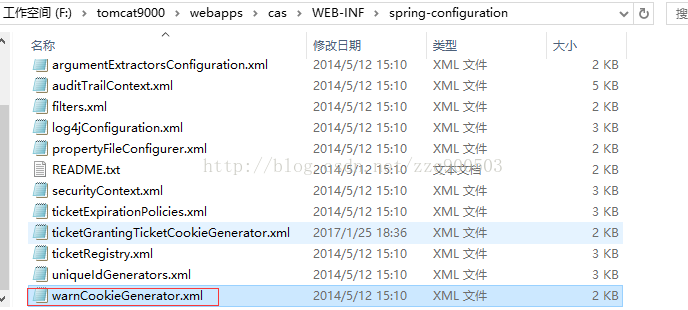
参数p:cookieMaxAge="-1"，简单说是COOKIE的最大生命周期，-1为无生命周期，即只在当前打开的IE窗口有效，IE关闭或重新打开其它窗口，仍会要求验证。可以根据需要修改为大于0的数字，比如3600等，意思是在3600秒内，打开任意IE窗口，都不需要验证。

这里把 cookieSecure修改为false就行了



**修改三 warnCookieGenerator.xml修改p:cookieSecure="false"**

我们在tomcat的webapp中找到cas/WEB-INF/spring-configuration/warnCookieGenerator.xml



<bean

id="warnCookieGenerator" class="org.jasig.cas.web.support.CookieRetrievingCookieGenerator"

p:cookieSecure="true"

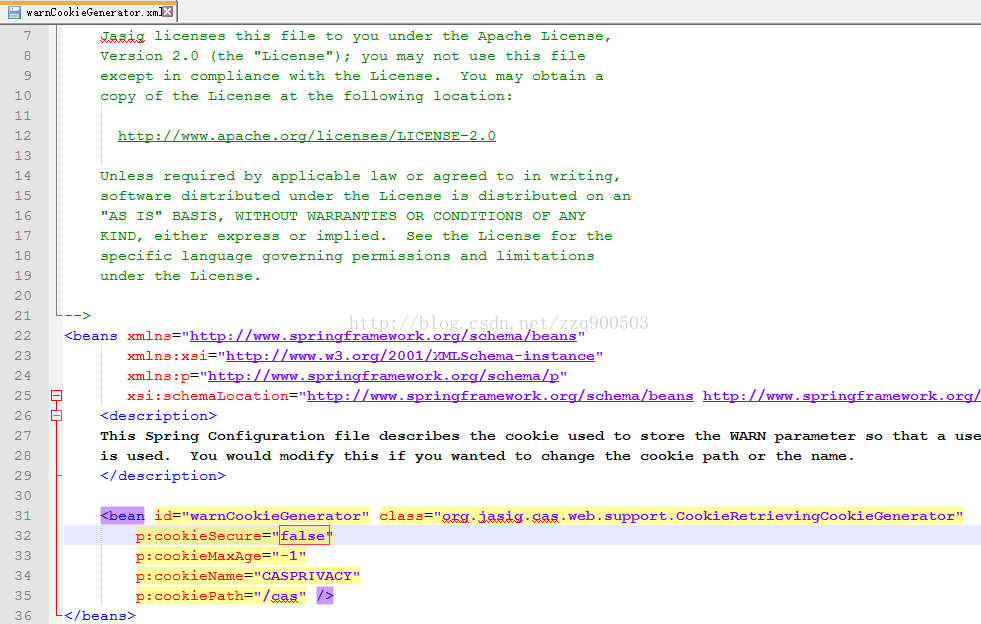
p:cookieMaxAge="-1"

p:cookieName="CASPRIVACY"

p:cookiePath="/cas" />

参数p:cookieSecure="true"，同理为HTTPS验证相关，TRUE为采用HTTPS验证，FALSE为不采用https验证。

这里把 cookieSecure修改为false就行了。

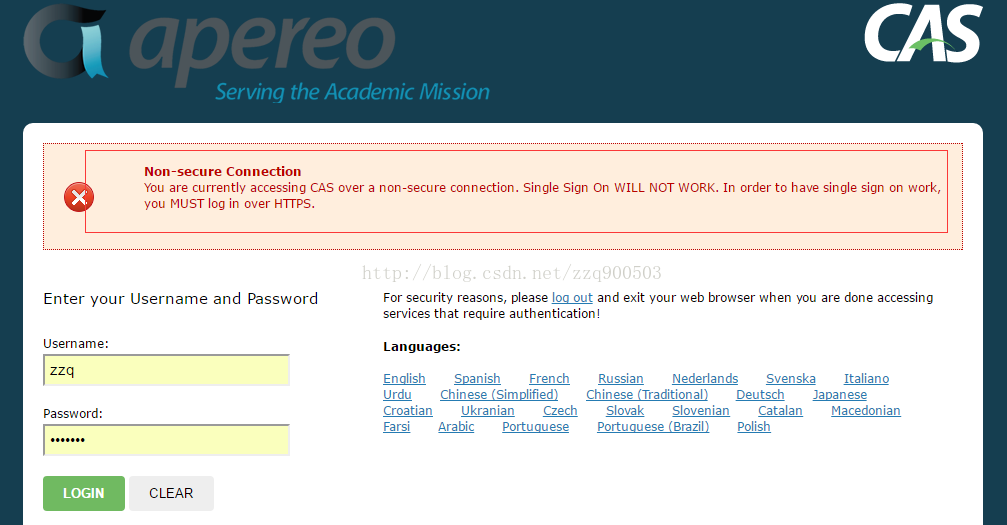


三个地方都修改好后我们就可以启动tomcat了，这时候https的方式已经取消，可以使用http的方式访问了。

**去掉登录页面上的提示**

我们去掉https验证之后 保证了我们在cas server和cas client 使用http访问也能进行正常交互。

但是原生的cas server页面上的提示是不会根据设置自动消失的。只能我们手动去除。



如果需要去除的话吗，步骤如下:

配置完http方式访问之后页面上的 提示还是存在的，如果我们之后会对登录界面样式完全改版，所以可以不用管它。如果还是需要把它去掉的话，cas统一认证的登陆页面位于：cas目录/WEB-INF/view/jsp/default 文件夹里，其中ui/casLoginView.jsp为登陆页面。我们找到这段代码删掉即可。

<c:if test="${not pageContext.request.secure}">

  <div id="msg" class="errors">

    <h2>Non-secure Connection</h2>

    <p>You are currently accessing CAS over a non-secure connection.  Single Sign On WILL NOT WORK.  In order to have single sign on work, you MUST log in over HTTPS.</p>

  </div>

</c:if>

