**CAS集成手册(java版)**

**东软集团股份有限公司**

**政府事业部**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **修改记录** | | | | | |
|  | | | | | |
| 版本号 | 变更控制报告  编号 | 更改条款及内容 | 更改人 | 审批人 | 更改日期 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

目录

[一、设计思想 4](#_Toc283276337)

[二、实现逻辑 4](#_Toc283276338)

[三、编码实现 4](#_Toc283276339)

[1、 JDK版本要求 4](#_Toc283276340)

[2、 客户端JAR包 5](#_Toc283276341)

[3、 增加并配置casFilterConfig.xml 5](#_Toc283276342)

[4、 修改登录入口 6](#_Toc283276343)

# 一、设计思想

CAS（Central Authentication Service – 中心认证服务）的目的就是使分布在内部各个不同异构系统的认证工作集中在一起，通过一个公用的认证系统统一验证用户身份的合法性。通过CAS认证的用户将获得CAS颁发的一个证书，使用这个证书，用户可以在承认CAS证书的各个系统上自由穿梭访问，不需要再次的登录认证。

# 二、实现逻辑

CAS客户端在认证过程中的实现逻辑：

1）以filter的形式，对B/S结构应用系统资源进行过滤保护，可以保护所有资源，也可以保护单一资源。

2）获得CAS Server颁发的ServiceTicket，并凭此ST从CAS Server上取得登录用户信息。

3）为第三方应用提供开发接口，使得受保护的应用能够根据CAS认证信息进行正确的登录。

# 三、编码实现（新系统集成）

通过CAS与第三方应用系统（以下统一简称为【**集成系统】**）集成时，应用系统开发商需要完成以下四步工作：

## JDK版本要求

程序使用的JDK必须是1.5及以上版本。

## 客户端JAR包

将我公司提供的CAS Client Jar包（neusoft-cas-4.0.0.jar），以及另外两个 Jar 包 cas-client-core-3.2.1.jar ，commons-logging-1.1.jar放在集成系统的WEB-INF/lib目录下。

## 增加并配置casFilterConfig.xml

新增配置文件casFilterConfig.xml，该文件放在集成系统的WEB-INF/classes目录下，内容示例如下：

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <config>  <loginServer> https://sso.scut.edu.cn/cas/</loginServer>  <validateServer>https://sso.scut.edu.cn/cas/</validateServer>  <serverName>http://localhost:8080</serverName >  <notForceAuthUrls>  <url-pattern>\.js$</url-pattern>  <url-pattern>\.xsl$</url-pattern>  <url-pattern>\.xml$</url-pattern>  <url-pattern>\.gif$</url-pattern>  <url-pattern>\.jpg$</url-pattern>  <url-pattern>\.bmp$</url-pattern>  <url-pattern>\.css$</url-pattern>  <url-pattern>\.swf$</url-pattern>  </notForceAuthUrls>  </config> |

各节点代表含义说明：

<loginServer>指的是CAS Server的登陆URL；

<validateServer>指的是CAS Server的验证URL（需要这个配置来告诉CAS Client获取Service Ticket后发送给这个地址进行验证从而取得用户信息）。对于目前我公司的CAS Server版本来说与loginServer是相同的。

<serverName>指的是当前集成系统所在的服务器和端口号，服务器可以是机器名、域名或ip，建议使用域名。端口不指定的话默认是80**（此处不能附加http://和应用名称）**。

<notForceAuthUrls>下的<url-pattern>指定了不需要CAS Filter进行过滤的资源，当过滤所有资源的时候使用。

## 修改登录入口

统一使用CAS登录入口，在应用系统WEB-INF目录下的web.xml文件中，加入CAS Filter相关的过滤器配置，增加内容如下：

|  |
| --- |
| <filter>  <filter-name>CAS-Filter</filter-name>  <filter-class>  com.neusoft.education.tp.sso.client.filter.DefaultCASFilter  </filter-class>  </filter>  <filter-mapping>  <filter-name>CAS-Filter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping> |

集成系统登录入口代码修改：

* 引入cas认证类

引入下面的类：

|  |
| --- |
| org.jasig.cas.client.authentication.AttributePrincipal |

* 修改原登录入口的获取用户账号方法，使用下面的方法获取登录用户

|  |
| --- |
| AttributePrincipal principal = (AttributePrincipal)request.getUserPrincipal();  String userID = principal.getName();  或者  String userName=request.getRemoteUser(); |

* 修改登录验证

根据上面的方法获取用户账号后，集成系统需要修改验证方法，不再验证密码信息，只判断该用户是否存在：

如果存在，则实现集成系统的登录（例如往session中写入身份信息、权限信息等）；

如果不存在，按集成系统逻辑处理。

上述配置封闭了原有登录入口，统一使用CAS登录入口。如果希望保留原有登录入口，新增加应用访问入口，新增入口与cas认证集成。在集成系统WEB-INF目录下的web.xml文件中的配置内容如下所示：

|  |
| --- |
| <filter>  <filter-name>CAS-Filter</filter-name>  <filter-class>  com.neusoft.education.tp.sso.client.filter.DefaultCASFilter  </filter-class>  </filter>  <filter-mapping>  <filter-name>CAS-Filter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping> |

如果希望保留原始的登录页面，并且登录成功之后可以跳转到指定的URL地址，那么需要在用于登录的JSP文件中做一下简单的处理，接收一个 url 地址，然后重定向到这个接收的url地址上即可，如下：

|  |
| --- |
| <%  String url = request.getParameter("url");  if(url!=null && !"".equals(url)){  response.sendRedirect(java.net.URLDecoder.decode(url,"UTF-8"));  }  %> |

注：地址栏往登录页面跳转的时候url后面的参数需要用 encodeURIComponent 函数转换一下，如：

|  |
| --- |
| encodeURIComponent("http://www.neusoft.com?a=111&b=2222");  转后为：http%3A%2F%2Fwww.neusoft.com%3Fa%3D111%26b%3D2222 |

## 获取用户属性

认证系统只提供账号、姓名、学号或教工号三个基本信息，需要其他信息的话，需要特别说明并且向网络中心提交申请。

|  |
| --- |
| <%  //获取账号信息  AttributePrincipal principal = (AttributePrincipal)request.getUserPrincipal();  String loginName = principal.getName();  // 以下代码取得more信息  Map attributes = principal.getAttributes();  System.out.println(attributes);  %> |

属性说明

|  |
| --- |
| ID\_NUMBER：账号  USER\_NAME：姓名  EDUPERSONSTAFFID：教工号  EDUPERSONSTUDENTID：学号  UNIT\_UID：单位号  UNIT\_NAME：单位名称  CARD\_ID：身份证号  USER\_SEX：性别  EMAIL：邮箱 |

# 集成过程报错信息及解决办法

## 页面500错误，后台报错信息如下：

|  |
| --- |
| <cas:serviceResponse xmlns:cas='http://www.yale.edu/tp/cas'>  <cas:authenticationFailure code='INVALID\_TICKET'>  未能够识别出目标 &#039;ST-34-0kR4ZnT1nmogpSDlDWhR-tpass&#039;票根  </cas:authenticationFailure>  </cas:serviceResponse> |

问题原因1：票根有效期时间较短，如果长时间不校验，就会提示上面错误；

问题原因2：票根只能应用一次，多次使用会提示上面错误；

请检查网络延迟问题，与是否多次使用同一个票根校验。

正常响应信息如下：

|  |
| --- |
| <sso:serviceResponse xmlns:sso='http://www.yale.edu/tp/cas' xmlns:cas='http://www.yale.edu/tp/cas'>  <sso:authenticationSuccess>  <sso:user>09901</sso:user>  <sso:attributes>  <sso:attribute name="id\_number" type="java.lang.String" value="09904"></sso:attribute>  <sso:attribute name="user\_name" type="java.lang.String" value="测试账号"></sso:attribute>  <sso:attribute name="eduPersonStaffID" type="java.lang.String" value="20170101"></sso:attribute>  <sso:attribute name="eduPersonStudentID" type="java.lang.String" value=""></sso:attribute>  </sso:attributes>  <cas:attributes>  <cas:ID\_NUMBER>09904</cas:ID\_NUMBER>  <cas:USER\_NAME>测试账号</cas:USER\_NAME>  <cas:eduPersonStaffID >20170101</cas:eduPersonStaffID >  <cas:eduPersonStudentID></cas:eduPersonStudentID>  <cas:checkAliveTicket>TGT-09901-26-3rcvDrIzG1JdmktMclJxmLQVEKAao0OuzncuDsCLFy7xLt5omF-cas</cas:checkAliveTicket>  <cas:serviceId>5471633887232</cas:serviceId>  </cas:attributes>  </sso:authenticationSuccess>  </sso:serviceResponse> |

## 票根'ST-1-SSAyf0nFMvmVBvEBSXyg-tpass'不符合目标服务

问题分析：使用A服务地址换取的票根，只能应用于A服务地址；如果校验票根的时候地址发生变化，则出现上述错误提示。

## 服务器连接不到域名

请查看网络连接是否可以连通认证服务器

## 集成过程中出现某个方法无法找到

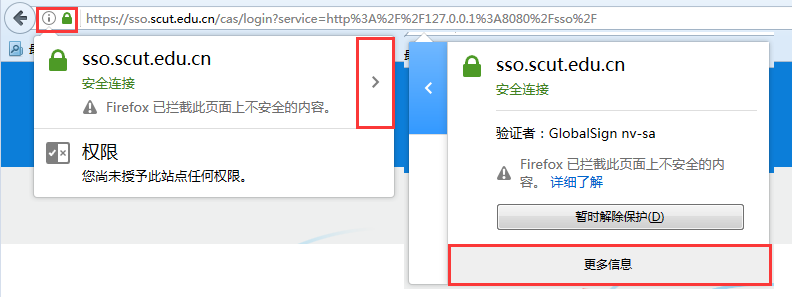
请查看jdk版本是否低于1.5版本

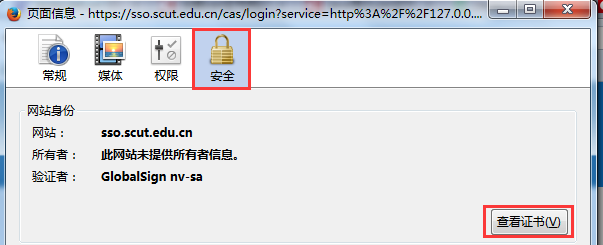
## HTTP Status 500 - javax.net.ssl.SSLHandshakeException

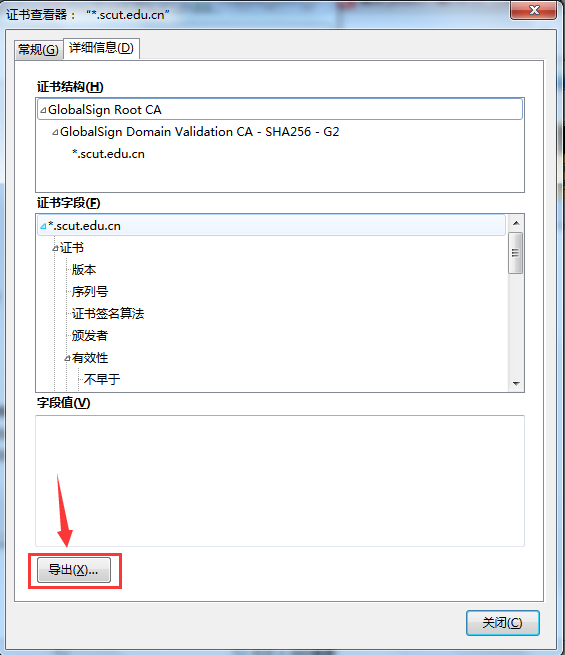
HTTP Status 500 - javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target

解决步骤：

1. 导出证书







将导出的证书导入到服务器的jdk中

将证书放在${JAVA\_HOME}/jre/lib/security目录中，同时进入到该目录，执行下面的命令（注意：一定要是服务器中间件（如tomcat）运行时的jdk）

|  |
| --- |
| keytool -import -alias scuteducn -keystore cacerts -file ${JAVA\_HOME}/jre/lib/security/-scuteducn.crt |

此时会提示要输入密码，输入changeit，之后输入y

以后需要更新证书时，先删除原来的证书，然后导入新的证书

keytool -delete -alias scuteducn -keystore cacerts

keytool -import -alias scuteducn -keystore cacerts -file ${JAVA\_HOME}/jre/lib/security/-scuteducn.crt