项目编号	INFO-115-C01
文档编号	TR-REC-034

中国科学院数据应用环境建设与服务

跨域用户认证接口规范

（征求意见稿）

中国科学院数据应用环境建设与服务 项目组

2009 年 6 月

目　　次

跨域用户认证接口规范

1 范围

本规范规定了中国科学院数据应用环境建设与服务项目内跨域用户认证接口采用的协议，连接方式，调用参数以及数据的返回格式。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB 18030-2005 信息技术　中文编码字符集

3 术语和定义

下列术语和定义适用于本规范。

- 资源 resource

可以被标识的实体对象或服务。

在本规范准中，特指可被标识的数据集、数据或服务。

- 数据集　dataset

由相关数据组成的可标识集合。一个数据集可能是一个较小的数据集合，在物理上或逻辑上位于一个较大的数据集之内；反之，一个数据集也可能由若干数据集组成，是这些子数据集的父数据集。

本规范中，数据集指中国科学院数据应用环境建设与服务项目的主题数据库、参考数据库、专题数据库、专业数据库及其各级子库。

- 服务 service

本规范中，服务是指数据应用环境建设与服务项目中数据集满足应用系统或者人的需要时，与之发生的“直接”接触行为及有关结果。

- 标识符 identifier

用于标识数据对象或服务项目的一组字符。

4 符号与缩略语

URI	统一资源标识符（Uniform Resource Identifier）
URL	统一资源定位符（Uniform Resource Locator）
XML	可扩展标记语言（Extensible Markup Language）

5 数据格式定义

5.1 接口的编码方式及响应格式

5.1.1 接口编码方式

接口响应的数据基于 xml 语言格式，编码方式采用自定义编码的方式（常用编码方式包括：UTF-8、GBK、GB2312、ISO-8859-1 等），接口响应数据所使用的编码方式必须在返回 XML 头信息的 encoding 属性中说明。

例如：

```
<?xml version="1.0" encoding="UTF-8" ?>
```

5.1.2 接口响应格式

```
<?xml version="1.0" encoding="UTF-8" ?>
<response>
    <head>
        <!--response 头信息-->
```

```
    <code>请求服务响应的状态码</code>
    <message>相关信息</message>
  </head>
  <body>
    <!--response 具体信息，接口返回的信息全部封装在 body 中-->
  </body>
</response>
```

5.1.3 接口响应请求状态码

以下状态码只列出接口中遇到的普遍问题，可根据数据库建库规范中规定的范围增加。

通用状态码	相关说明
200	服务调用成功
400	请求格式错误
401	未授权访问
402	不可识别的 verb
403	服务器拒绝访问
404	指定的资源未找到
408	请求超时
500	服务器内部错误

响应状态码返回格式：

```
<?xml version="1.0" encoding="UTF-8" ?>
<response>
  <head>
    <code>响应请求状态码</code>
    <message>响应请求状态码的描述</message>
  </head>
  <body>
    <!—接口返回的具体数据-->
  </body>
</response>
```

6 接口规范

6.1 采用协议

HTTP1.1（超文本传输协议）

超文本传输协议（HTTP）是一种为分布式，合作式，超媒体信息系统。它是一种通用的，无状态（stateless）的协议，除了应用于超文本传输外，它 也可以应用于诸如名称服务器和分布对象管理系统之类的系统，这可以通过扩展它的请求方法，错误代码和报头来实现。HTTP 的一个特点是数据表现形 式是可输入的和可协商性的，这就允许系统能被建立而独立于数据传输。

HTTP 是一个客户端和服务器端请求和应答的标准（TCP）。客户端是终端用户，服务器端是网站。通常，由 HTTP 客户端发起一个请求，建立一个到服务器指定端口（默认是 80 端口）的 TCP 连接。HTTP 服务器则在那个端口监听客户端发送过来的请求。 一旦收到请求，服务器（向客户端）发回一个状态行，比如"HTTP/1.1 200 OK"，和（响应的）消息，消息的消息体可能是请求的文件、错误消息、或者其它一些信息。

6.2 接口安全

采用基于 IP 地址的身份验证方式。在服务调用的过程中，服务提供者获取调用者的 ip 地址，在本地保存的授权访问 ip 地址列表中查询，认证通过执行服务返回数据，不通过则拒绝服务。

认证流程如下图所示：

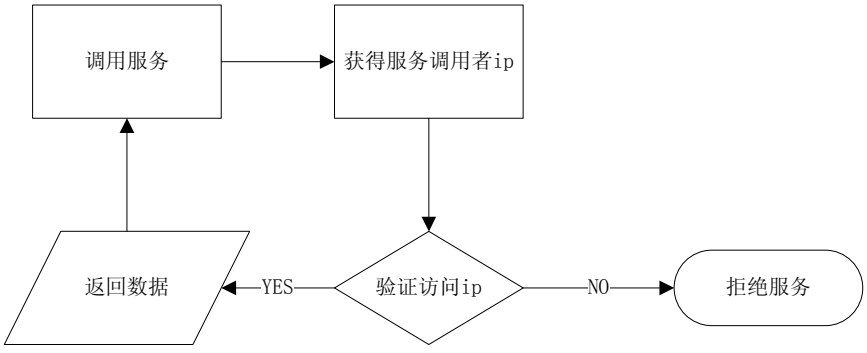

图 6-1 基于 ip 地址的身份认证方式

拒绝服务返回数据格式：

```
<?xml version="1.0" encoding="UTF-8" ?>
<response>
    <head>
        <code>403</code>
        <message>服务拒绝访问-ip 地址认证未通过</message>
    </head>
    <body></body>
</response>
```

6.3 连接方式

POST 方式（必须实现）：

POST 方法被用于请求源服务器接受请求中的实体作为请求资源的一个新的从属物。

POST 方法的实际功能是由服务器决定的，并且经常依赖于请求 URI（Request-URI）。POST 提交的实体是请求 URI 的从属物，就好像一个文件从属于一个目录，一篇新闻文章从属于一个新闻组，或者一条记录从属于一个数据库。

POST 方法的响应是可缓存的。

GET 方式（可选）：

GET 方法意思是获取被请求 URI（Request-URI）指定的信息（以实体的格式）。如果请求 URI 涉及到一个数据生成过程，那么这个生成的数据应该被作为实体在响应中返回，但这并不是过程的资源文本，除非资源文本恰好是过程的输出。

GET 请求的响应是可缓存的。

6.4 技术实现

接口采用 OpenURL 技术实现，OpenURL 技术规范详见附录 3.1 章节（OpenURL）。

接口语法格式说明：

http://url/service?[query]

[?]号前面为科学数据库参建单位提供的服务地址，需要在资源注册系统(http://rsr.csdb.cn)中注册。只有注册的服务科学数据中心才可以进行访问。

[query]部分包括多组参数名称与参数值，其中最主要的 verb 参数，定义了访问服务的具体名称见接口列表中的接口名称。

6.5 接口列表

本标准中接口列表如下：

接口提供方	接口类型	接口名称	接口描述
数据中心	用户认证与授权	usrGetUser	获取用户信息接口

6.6 科学数据中心开放接口

6.6.1 用户认证与授权接口

6.6.1.1 usrGetUser

接口说明：

用户授权接口，根据用户唯一标识获取用户基本信息。

请求参数说明：

verb：usrGetUser

userid：用户唯一标识

调用示例：

http://url/service?verb=usrGetUser&userid=user@163.com

响应格式说明：

<user>

<!—响应信息包括除密码外的所有信息-->

<userid>用户唯一标识</userid>

<email>电子邮件地址</email>

<unit>所在单位</unit>

<date>注册时间</date>

<ip>注册 ip</ip>

```
</user>
```

响应格式示例：

```
<user>
    <userid>test@163.com</userid>
    <email>test@163.com</email>
    <name>注册用户真实姓名</name>
    <occupation>科研院所研究人员</occupation>
    <unit>中国科学院计算机网络信息中心</unit>
    <address>北京市海淀区</address>
    <phone>(010)8123****</phone>
    <date>2009-01-01 17:38:46</date>
    <ip>159.226.3.***</ip>
</user>
```

附录 A （资料性附录） OpenURL

参考地址：http://en.wikipedia.org/wiki/OpenURL

语法格式：

OpenURL：：= BASE-URL ‘?’ QUERY

BASE-URL：基础 URL，服务提供方的 URL 地址

QUERY：查询，包含 ORIGIN-DESCRIPTION（参数名称），OBJECT-DESCRIPTION（参数值）两部分。

附录 B 跨域用户单点登录客户端配置

1 系统环境

1. 浏览器：IE8、FireFox
2. CAS Client：windows XP 或 linux 服务器
3. 应用服务器：tomcat6.0.20
4. JDK1.6

2 相关软件下载及安装

1. 安装 JDK:下载 JDK1.6 的 linux 版本。

   上传 JDK 至 linux 服务器，在 jdk 目录下安装 JDK,其是是一个解压过程。
2. 安装 tomcat:官网下载 apache-tomcat-6.0.20.tar.gz（linux 版本）

   上传 tomcat 至 linux 服务器，在对应目录下用命令：

   tar zxvf apache-tomcat-6.0.20.tar.gz

   解压 tomcat
3. 配置环境变量

   编辑~/.bashrc 文件里面设置环境变量

   cd ~

   ls -a

   vi .bashrc

   if [-f /etc/bashrc]; then

   . /etc/bashrc

   fi

   export JAVA_HOME=JDK 路径

   export CLASSPATH=$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib

   export PATH=$JAVA_HOME/bin:$JAVA_HOME/jre/bin:$PATH:$HOMR/bin

   export CATALINA_BASE=tomcat 路径

export CATALINA_HOME= tomcat 路径

4. 测试环境

到 tomcat 的 bin 目录下面启动 tomcat ./startup.sh

用 netstat -ln|grep 8080 查看 8080 端口

输入 http://ip:8080 是否可以看到 tomcat 首页。

如果没有看到修改防火墙规则，开发 8080 端口。

service iptables stop

service iptables start

5. Windows 下 JDK 和 tomcat 请自行安装。

3 配置 CAS 客户端（JAVA）

1. CAS 官网下载 cas-client-java-2.1.1.zip，将编译好的类文件打包为 casclient.jar、把 casclient.jar、commons-logging-api.jar 拷贝到应用的 WEB-INFO/lib 目录下。同时下载 cas-client-3.1.9-release.zip，把 cas-client-core-3.1.9.jar 放置于 WEB-INFO/lib 目录。
2. 为应用程序添加 ServletFilter 实现单点登录认证检查。

打开 WEB-INFO/web.xml

配置单点登录

<filter>

```
<filter-name>CAS Filter</filter-name>
<filter-class>edu.yale.its.tp.cas.client.filter.CASFilter</filter-class>
<init-param>
    <param-name>edu.yale.its.tp.cas.client.filter.loginUrl</param-name>
<param-value>
    https://服务器名称(eg:passport.csdb.cn):8443/cas/login
</param-value>
</init-param>
<init-param>
<param-name>edu.yale.its.tp.cas.client.filter.validateUrl</param-name>
<param-value>
    https://服务器名称(eg:passport.csdb.cn):8443/cas/serviceValidate
</param-value>
</init-param>
<init-param>
<param-name>edu.yale.its.tp.cas.client.filter.serverName</param-name>
<param-value>客户端名称:port</param-value>
</init-param>
</filter>
<filter-mapping>
    <filter-name>CASFilter</filter-name>
    <url-pattern>登录action(eg:login.do 或者 index.jsp)</url-pattern>
</filter-mapping>
```

配置单点退出

```
<listener>
```

```
    <listener-class>org.jasig.cas.client.session.SingleSignOutHttpSessionL
    istener</listener-class>
</listener>
<filter>
    <filter-name>CAS Single Sign Out Filter</filter-name>
    <filter-class>org.jasig.cas.client.session.SingleSignOutFilter</filter-cl
    ass>
</filter>
<filter-mapping>
    <filter-name>CAS Single Sign Out Filter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

注意事项：web.xml 中的 filter 要注意先后顺序，CAS Single Sign Out Filter 相关配置要放在所有配置 Filter 的前面，同时单点退出的<filter-mapping>也要放在所有<filter-mapping>之前。

具体应用的退出方式：单击退出操作使其跳转到统一认证服务器，有认证服务器统一处理。

跳转地址：https://服务器名称:8443/cas/logout

3. 在客户端的 JVM 里导入信任的认证服务器导出的证书

   通过运行 cmd 命令进$JAVA_HOME/jre/bin 目录，运行

   keytool -import -keystore $JAVA_HOME/jre/lib/security/cacerts -file server.crt -alias tomcat

   JVM 路径可以替换为绝对路径

4 测试配置

1. 在客户端输入 http://客户端:port/应用名称。
2. 单击登录跳转到统一认证登录页面，输入注册的用户名和密码，登录后转到具体应用。

3. 在输入另外的应用，可以直接取到用户名，而不需要在重新登录。

   取用户名方式：

   引入 edu.yale.its.tp.cas.client.filter.*包

   session.getAttribute(CASFilter.CAS_FILTER_USER)

4. 在应用中单击退出，可以注销已登录用户。
5. 单击注册按钮，跳转至 https://cas.csdb.cn:8443/cas/reg01.jsp 进行统一注册。

5 具体应用中角色处理

1. 如果用户直接访问应用，而未触发登录操作，此时用户以匿名身份访问，可浏览无需登录页面。
2. 如果用户触发登录操作，而应用系统无角色划分，可以允许登录用户访问相关页面，用户有统一认证系统管理。
3. 如果用户触发登录操作，而且应用系统有角色划分，应用可根据用户名进行授权处理，根据用户权限不同，访问不同页面。

   具体实现：在登录时，根据返回的用户名，把用户名插入到本地应用中，有应用进行授权处理。
4. 每个具体应用必须注册一个具有管理员权限的用户，应用对此用户进行特殊处理，此用户可以对应用的其他用户就行权限管理。