# 介绍

统一身份认证是一套完整独立、高效稳定、安全可靠的集中身份认证和分级授权管理、单人多角色管理平台。其必须提供广泛而灵活的认证服务与接口、信息采集与信息查询功能，实现单点登录服务以确保用户身份认证的准确性和便利性，实现各应用系统用户名和口令的统一；

# LDAP

使用的判断用户是否合法方式为用户和密码验证法, 在用户与服务器的通信过程中, 采用一种方式去确认对方用户的身份或权限, 然后再对用户的不同身份进行分配其访问控制权限。

结合我校现有系统, 用LDAP技术可以解决不同子系统统一身份认证的问题, 利用LDAP目录服务集中存储用户身份数据, 实现VPN、上网、无线认证以及应用等认证。

LDAP认证过程

LDAP是轻量目录访问协议(Lightweight Directory Access Protocol) ；LDAP认证是通过WSS3.0加上轻量目录LDAP协议搭建的种认证方式，使用https加密传输，主要用于做文档管理。LDAP认证就是把用户数据放在LDAP服务器上，通过LDAP服务器上的数据对用户进行认证处理。

LDAP实现原理：每一个登陆，连接请求去发送本地的用户、密码给LDAP服务器，然后在LDAP服务器上进行匹配，然后判断是否可以通过认证。

LDAP优点：

1).LDAP数据库对读操作进行优化的种数据库，在读写比例大于7比1的情况下，LDAP会体现出高的性能。

2).更灵活添加数据类型，LDAP是根据schema的内容定义各种属性之间的从属关系及匹配模式的。

例如：在传统的结构化数据库mysql中添加一个字段，就需要在用户表中添加一个字段。但是在数据量

极大的时候是很耗时间的，效率低，用户体验差，但是LDAP只需要在Schema中加入新的属性，不会

由于用户的属性增多而形象查询性能。

3).LDAP是个开放的标准协议，不同于一般的SQL数据库，LDAP的客户端是跨平台的，方便简洁。

4).在存储上LDAP是以树形结构存储数据，任何一个分支都可以单独在服务器中进行分布式管理，

不仅有利于服务器的负载均衡，还方便做跨区域的服务器部署。

5).LDAP支持强认证方式，可以达到很高的安全级别，根据UTF-8编码。

LDAP接口主要面向瞬时认证并发要求非常高的应用系统，最典型的就是高校的VPN、无线网认证。利用LDAP接口实现应用系统认证集成以后，可实现高效的统一认证。

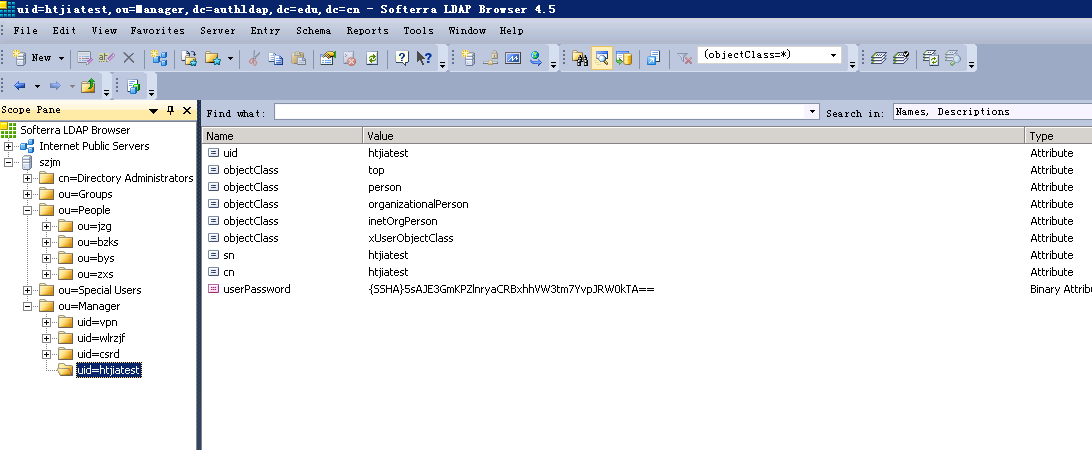
uid=htjiatest,ou=Manager,dc=authldap,dc=edu,dc=cn

ldap://192.168.12.60:389/uid=htjiatest,ou=Manager,dc=authldap,dc=edu,dc=cn

DC (Domain Component):所在控制域

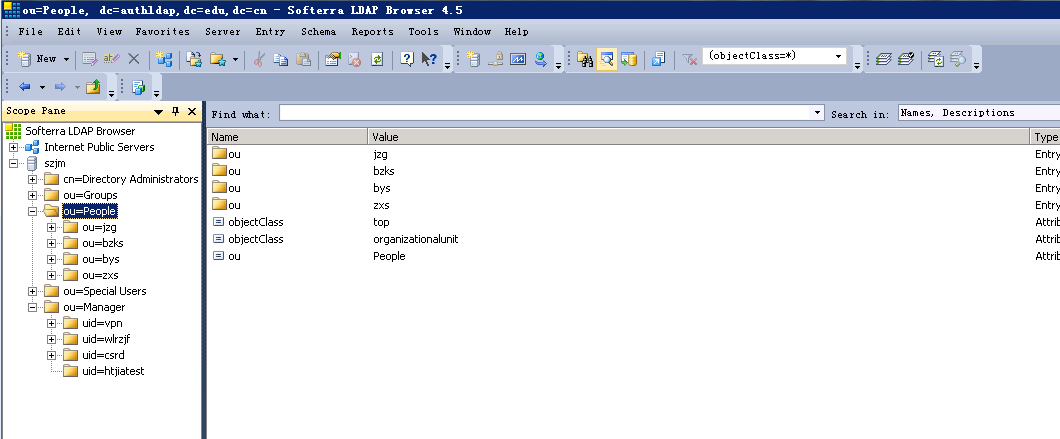
OU (Organizational Unit):组织单元

CN (Common Name):通用名称



在校生

ldap://192.168.12.60:389/ou=zxs,ou=People,dc=authldap,dc=edu,dc=cn



系统后台维护部分





LDAP教职工同步过程

select \* from V\_JZG --差异视图

(基本表和差异表比对以后的表)

|  |
| --- |
| CREATE OR REPLACE VIEW V\_JZG  (id, name, password, sfzx, actiontype)  AS  SELECT t.id, t.name, t.password, '1', 'add'  FROM (SELECT a.gh AS id,  a.XM AS name,  '88888888' AS password  FROM t\_hr\_jzg a  where a.gh <> 'ampadmin') t  WHERE NOT EXISTS (SELECT c.id FROM V\_JZG\_BASIC c WHERE t.id = c.id) |

select \* from V\_JZG\_BASIC --基本表

(ldap里面一致的表)

select \* from t\_hr\_jzg --标准表(数据标准):

(人事系统同步过来的标准表)



# CAS

将学校的各个业务系统与身份管理平台连接，通过身份管理平台实现用户身份认证和单点登录功能，这个过程即应用系统与身份管理平台的集成。

(1)认证接口：这是各个应用系统与身份管理平台集成的最主要的方式。各应用将认证接口的客户端开发包集成在各应用之中，替换自身原有独立的身份认证功能，通过身份管理平台实现身份认证和单点登录过程。该接口目前NET(2.0+)语言和平台的应用程序。

(2)LDAP接口：对于选课选这样的高并发应用，我们提供LDAP接口，以满足认证的性能需求。该接口直接通过LDAP向应用系统提供认证服务，但是牺牲了单点登录功能。

代理认证配置完后，均需至统一身份管理平台授权访问。先登录身份认证管理平台，在认证管理的认证应用模板，添加需代理认证的应用；添加完应用后，需给相应的组或者帐号授权，允许其访问该应用。

管理界面首先，认证服务添加应用，如下图所示：



然后添加用户授权



代码部分实现原理：

1. 拷贝提供的jar
2. 修改web.xml文件
3. 获取用户信息
4. 集成应用退出
5. 认证接口工作过程



# OAuth 2.0协议

## 名词定义

（1） Third-party application：第三方应用程序，本文中又称"客户端"（client），即本例程序“Hello World”。

（2）HTTP service：HTTP服务提供商，本文中简称"服务提供商"，即例子中的IDS。

（3）Resource Owner：资源所有者，本文中又称"用户"（user）。

（4）User Agent：用户代理，本文中就是指浏览器。

（5）Authorization server：认证服务器，即服务提供商专门用来处理认证的服务器。

（6）Resource server：资源服务器，即服务提供商存放用户生成的资源的服务器。它与认证服务器，可以是同一台服务器，也可以是不同的服务器。

OAuth的作用就是让"客户端"安全可控地获取"用户"的授权，与"服务商提供商"进行互动。

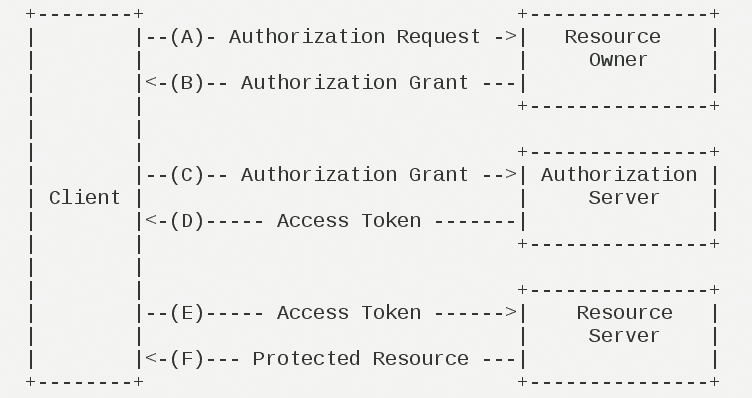
## OAuth的思路

OAuth在"客户端"与"服务提供商"之间，设置了一个授权层（authorization layer）。"客户端"不能直接登录"服务提供商"，只能登录授权层，以此将用户与客户端区分开来。"客户端"登录授权层所用的令牌（token），与用户的密码不同。用户可以在登录的时候，指定授权层令牌的权限范围和有效期。

"客户端"登录授权层以后，"服务提供商"根据令牌的权限范围和有效期，向"客户端"开放用户储存的资料。

## 运行流程

OAuth 2.0的运行流程如下图，摘自RFC 6749。



Resource Owner:资源拥有者

Authorization Servicer:验证服务器

Resource Service :资源服务器

（A）用户打开客户端以后，客户端要求用户给予授权。

（B）用户同意给予客户端授权。

（C）客户端使用上一步获得的授权，向认证服务器申请令牌。

（D）认证服务器对客户端进行认证以后，确认无误，同意发放令牌。

（E）客户端使用令牌，向资源服务器申请获取资源。

（F）资源服务器确认令牌无误，同意向客户端开放资源。

## 客户端的授权模式

客户端必须得到用户的授权（authorization grant），才能获得令牌（access token）。OAuth 2.0定义了四种授权方式。

授权码模式（authorization code）

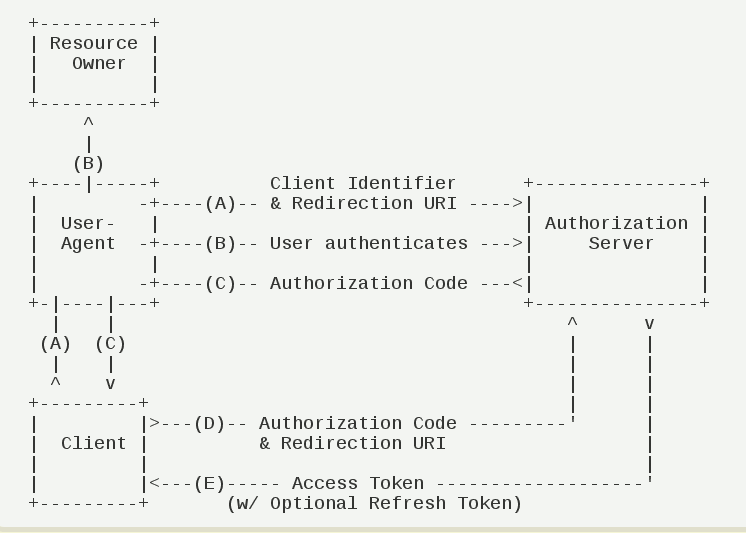
简化模式（implicit）

密码模式（resource owner password credentials）

客户端模式（client credentials）

## 授权码模式

授权码模式（authorization code）是功能最完整、流程最严密的授权模式。它的特点就是通过客户端的后台服务器，与"服务提供商"的认证服务器进行互动。



（A）用户访问客户端，后者将前者导向认证服务器。

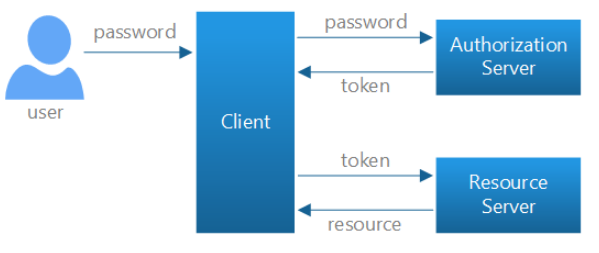
（B）用户选择是否给予客户端授权。

（C）假设用户给予授权，认证服务器将用户导向客户端事先指定的"重定向URI"（redirection URI），同时附上一个授权码。

（D）客户端收到授权码，附上早先的"重定向URI"，向认证服务器申请令牌。这一步是在客户端的后台的服务器上完成的，对用户不可见。

（E）认证服务器核对了授权码和重定向URI，确认无误后，向客户端发送访问令牌（access token）和更新令牌（refresh token）。

密码模式（resource owner password credentials）的流程：



这种模式的流程非常简单：

A 用户向客户端(third party application)提供用户名和密码。

B 客户端将用户名和密码发给认证服务器(Authorization server)，向后者请求令牌(token)。

C 认证服务器确认无误后，向客户端提供访问令牌。

D 客户端持令牌(token)访问资源。

# 参考文献

OAuth2.0 参考

<http://www.ruanyifeng.com/blog/2014/05/oauth_2_0.html>

<https://www.cnblogs.com/flashsun/p/7424071.html>