

# Using Fedora Directory Server and HTTP authentication via LDAP (htaccess and Apache) (Last Revision 06052007 written by Ashley Chew)

## Background Information

This document is on about the use of web authentication via LDAP especially to the Fedora Directory Server in terms of web configuration of Apache on a system level and htaccess on a user defined level. The only reason why I've written this document is when I was reading some people documentation some essential steps were effectively left out.

## Requirements

Now I'm assuming you have a functional Fedora Directory Server which is essentially an LDAP server (jhett.csse.uwa.edu.au as with my other documentation) and a Fedora Web Server (testweb.csse.uwa.edu.au running FC6) with essentially apache running.

## Instructions

Now for Apache (Also known as httpd service) to use the FDS LDAP for authentication, you have to a module compiled or shared module. The module which I would recommend is the **mod\_authnz\_ldap**, not mod\_auth\_ldap module. And the syntax is applicable to **mod\_authnz\_ldap** and not mod\_auth\_ldap.

Now lets check if the package is physically installed on Fedora Core 6 by doing this command

```
[root@testbox]# rpm -qa |grep -i mod_authz_ldap
mod_authz_ldap-0.26-7.1
```

As you can see the mod\_authz\_ldap package is installed, physically all the apache modules if it not compiled in and exist as a shared module it will be located here (This on a Fedora Core 6 Distribution)

```
[root@ testweb]# pwd
/usr/lib/httpd/modules
[root@ testweb]# ls
```

```

libphp5.so      mod_authn_file.so  mod_cern_meta.so  mod_ext_filter.so
mod_mime_magic.so  mod_speling.so
mod_actions.so   mod_authnz_ldap.so  mod_cgid.so      mod_file_cache.so
mod_mime.so      mod_ssl.so
mod_alias.so     mod_auth_pgsql.so   mod_cgi.so       mod_filter.so
mod_negotiation.so  mod_status.so
mod_asis.so      mod_authz_dbm.so    mod_dav_fs.so    mod_headers.so
mod_perl.so      mod_suexec.so
mod_auth_basic.so  mod_authz_default.so  mod_dav.so       mod_ident.so
mod_proxy_ajp.so   mod_unique_id.so
mod_auth_digest.so  mod_authz_groupfile.so  mod_dav_svn.so   mod_imagemap.so
mod_proxy_balancer.so  mod_userdir.so
mod_auth_kerb.so   mod_authz_host.so    mod_dbd.so       mod_include.so
mod_proxy_connect.so  mod_usertrack.so
mod_auth_mysql.so  mod_authz_ldap.so    mod_deflate.so   mod_info.so
mod_proxy_ftp.so   mod_version.so
mod_authn_alias.so  mod_authz_owner.so   mod_dir.so       mod_ldap.so
mod_proxy_http.so  mod_vhost_alias.so
mod_authn_anon.so  mod_authz_svn.so     mod_disk_cache.so  mod_log_config.so
mod_proxy.so
mod_authn_dbd.so   mod_authz_user.so    mod_dumpio.so    mod_log_forensic.so
mod_python.so
mod_authn_dbm.so   mod_autoindex.so     mod_env.so        mod_logio.so
mod_rewrite.so
mod_authn_default.so  mod_cache.so        mod_expires.so    mod_mem_cache.so
mod_setenvif.so
[root@ testweb]#

```

Which you can see the mod\_authz\_ldap.so which provides the LDAP authentication. Now if your missing the file or rpm install it first and restart apache by typing this.

```
/etc/init.d/httpd restart
```

Now although physically you have the files and package installed, you have to see if Apache has the module loaded into it, you can verify this by typing this.

```

[root@ testweb]# /usr/sbin/apachectl -M
Loaded Modules:
core_module (static)
mpm_prefork_module (static)
http_module (static)
so_module (static)
auth_basic_module (shared)
auth_digest_module (shared)
authn_file_module (shared)
authn_alias_module (shared)

```

Written by: Ashley Chew  
Email: ashley@csse.uwa.edu.au  
Last Revision: 05062007

authn\_anon\_module (shared)  
authn\_dbm\_module (shared)  
authn\_default\_module (shared)  
authz\_host\_module (shared)  
authz\_user\_module (shared)  
authz\_owner\_module (shared)  
authz\_groupfile\_module (shared)  
authz\_dbm\_module (shared)  
authz\_default\_module (shared)  
ldap\_module (shared)  
authnz\_ldap\_module (shared)  
include\_module (shared)  
log\_config\_module (shared)  
logio\_module (shared)  
env\_module (shared)  
ext\_filter\_module (shared)  
mime\_magic\_module (shared)  
expires\_module (shared)  
deflate\_module (shared)  
headers\_module (shared)  
usertrack\_module (shared)  
setenvif\_module (shared)  
mime\_module (shared)  
dav\_module (shared)  
status\_module (shared)  
autoindex\_module (shared)  
info\_module (shared)  
dav\_fs\_module (shared)  
vhost\_alias\_module (shared)  
negotiation\_module (shared)  
dir\_module (shared)  
actions\_module (shared)  
speling\_module (shared)  
userdir\_module (shared)  
alias\_module (shared)  
rewrite\_module (shared)  
proxy\_module (shared)  
proxy\_balancer\_module (shared)  
proxy\_ftp\_module (shared)  
proxy\_http\_module (shared)  
proxy\_connect\_module (shared)  
cache\_module (shared)  
suexec\_module (shared)  
disk\_cache\_module (shared)  
file\_cache\_module (shared)  
mem\_cache\_module (shared)

Written by: Ashley Chew  
Email: [ashley@csse.uwa.edu.au](mailto:ashley@csse.uwa.edu.au)  
Last Revision: 05062007

```
cgi_module (shared)
auth_kerb_module (shared)
mysql_auth_module (shared)
auth_pgsqldb_module (shared)
authz_ldap_module (shared)
perl_module (shared)
php5_module (shared)
proxy_ajp_module (shared)
python_module (shared)
ssl_module (shared)
dav_svn_module (shared)
authz_svn_module (shared)
Syntax OK
[root@testbox modules]#
```

As you can see authz\_ldap\_module is loaded as a shared object. If it isn't you have to specify it to load it up by editing the Apache configuration in /etc/httpd/conf/httpd.conf. Look under the section "Dynamic Shared Object (DSO) Support"

Insert this line below into the configuration and restart apache.

**LoadModule authnz\_ldap\_module modules/mod\_authnz\_ldap.so**

Check again to see if Apache has loaded the module by issuing the "/usr/sbin/apachectl -M" command again.

Now that you have that going I would advise you to look at the LDAP Directives for authentication which you can find here

[http://httpd.apache.org/docs/2.2/mod/mod\\_authnz\\_ldap.html](http://httpd.apache.org/docs/2.2/mod/mod_authnz_ldap.html)

But typically the attributes you will be playing with are these Directives

```
require valid-user
require ldap-user
require ldap-group
require ldap-dn
require ldap-attribute
require ldap-filter
```

Now usually you can use these directives on a system level by setting it in /etc/httpd/conf/httpd.conf, or as a user level by enabling .htaccess files. The only difference is that if is enabled at a system level, normal users cannot change the access it has to be changed by root user hence the user level via .htaccess. But the syntax used for either is the same.

Let say we wanted to allow access to people who has Unix/Linux Group ID of 2000. So the directive we would use is ldap-attribute and in the FDS schema the LDAP attribute for GID is gidNumber hence.

```
AuthType Basic  
Allow from all  
AuthBasicProvider ldap  
AuthName "Authentication via Linux Group GID Number"  
AuthLDAPUrl  
"ldap://jhett.csse.uwa.edu.au:389/ou=People,dc=csse,dc=uwa,dc=edu,dc=au"  
AuthzLDAPAuthoritative Off  
Require ldap-attribute gidNumber=2000
```

Another typical example is that you want to enable only certain users who don't belong to any particular group

```
AuthType Basic  
Allow from all  
AuthBasicProvider ldap  
AuthName "Authentication via User"  
AuthLDAPUrl  
"ldap://jhett.csse.uwa.edu.au:389/ou=People,dc=csse,dc=uwa,dc=edu,dc=au"  
AuthzLDAPAuthoritative Off  
Require ldap-user test01  
Require ldap-user test02  
Require ldap-user test03
```

The most useful one would be group access, be careful with this especially the members of the group has to be explicitly listed as members in the ldap group.

```
AuthType Basic  
Allow from all  
AuthBasicProvider ldap  
AuthName "Authentication via Linux Group"  
AuthLDAPUrl  
"ldap://jhett.csse.uwa.edu.au:389/ou=People,dc=csse,dc=uwa,dc=edu,dc=au"  
AuthzLDAPAuthoritative Off  
Require ldap-group cn=testgroup,ou=Groups,dc=csse,dc=uwa,dc=edu,dc=au
```

Now If you have LDAPs working on your box, ie in my case testweb.csse.uwa.edu.au I would enable the LDAPs connection instead of the standard LDAP connection which is pretty simple change ldap->ldaps and port number from 389->636 (Standard Defined ports for LDAP) ie

```
AuthType Basic  
Allow from all
```

**AuthBasicProvider ldap**

**AuthName "Authentication via Linux Group with LDAPs"**

**AuthLDAPUrl**

**"ldaps://jhett.csse.uwa.edu.au:636/ou=People,dc=csse,dc=uwa,dc=edu,dc=au"**

**AuthzLDAPAuthoritative Off**

**Require ldap-group cn=testgroup,ou=Groups,dc=csse,dc=uwa,dc=edu,dc=au**

Now those examples above are general syntax, as I mentioned before you can implement the directives either on a system level or user level. Now using the above Group Authentication example to implement this on a user level I would put this in the httpd.conf (or in the virtualhost which is sourced by httpd.conf).

**<VirtualHost 192.168.0.5>**

**ServerAdmin webmaster@csse.uwa.edu.au**

**DocumentRoot /home/projects/virtualseite/public\_html**

**ServerName virtualseite.csse.uwa.edu.au**

**ErrorLog /home/projects/virtualseite/weblogs/error\_log**

**CustomLog /home/projects/virtualseite/weblogs/access\_log common**

**<Directory "/home/projects/virtualseite/public\_html">**

**AllowOverride AuthConfig Limit FileInfo**

**Order deny,allow**

**Allow from all**

**</Directory>**

**</VirtualHost>**

Don't forget the **AllowOverride** directive in the httpd.conf as this controls the directives in .htaccess ie

# AllowOverride controls what directives may be placed in .htaccess files.

# It can be "All", "None", or any combination of the keywords:

# Options FileInfo AuthConfig Limit

Now I would create **/home/projects/virtualseite/public\_html/.htaccess** and put this.

**AuthType Basic**

**Allow from all**

**AuthBasicProvider ldap**

**AuthName "Authentication via Linux Group with LDAPs"**

**AuthLDAPUrl**

**"ldaps://jhett.csse.uwa.edu.au:636/ou=People,dc=csse,dc=uwa,dc=edu,dc=au"**

**AuthzLDAPAuthoritative Off**

**Require ldap-group cn=testgroup,ou=Groups,dc=csse,dc=uwa,dc=edu,dc=au**

This allows people to change .htaccess by themselves without any intervention from the system admin. But if you want to lock it in at the system level you would do this instead in httpd.conf (or in the virtualhost which is sourced by httpd.conf).

Written by: Ashley Chew  
Email: ashley@csse.uwa.edu.au  
Last Revision: 05062007

```
<VirtualHost 192.168.0.5>
  ServerAdmin webmaster@csse.uwa.edu.au
  DocumentRoot /home/projects/virtualseite/public_html
  ServerName virtualseite.csse.uwa.edu.au
  ErrorLog /home/projects/virtualseite/weblogs/error_log
  CustomLog /home/projects/virtualseite/weblogs/access_log
  <Directory "/home/projects/virtualseite/public_html">
    AuthType Basic
    Allow from all
    AuthBasicProvider ldap
    AuthName "Authentication via Linux Group with LDAPs"
    AuthLDAPUrl
    "ldaps://jhett.csse.uwa.edu.au:636/ou=People,dc=csse,dc=uwa,dc=edu,dc=au"
    AuthzLDAPAuthoritative Off
    Require ldap-group cn=testgroup,ou=Groups,dc=csse,dc=uwa,dc=edu,dc=au
  </Directory>
</VirtualHost>
```