Subversion with Apache and LDAP

The purpose of the **Definitive Guide** is to provide a single location for questions for Apache 2.0.x and 2.2.x, while also providing more depth about things to consider when building your Apache-based Subversion server using LDAP for authentication.

The Configuration

For those of you that just want to get to the point, where you can copy and paste and move on, here you go:

Example Apache 2.2.x Configuration Snippet

```
# Load Apache LDAP modules
LoadModule ldap_module modules/mod_ldap.so
LoadModule authnz_ldap_module modules/mod_authnz_ldap.so


# Load Subversion Apache Modules
LoadModule dav_svn_module     modules/mod_dav_svn.so # Use full path to
SUBVERSION_HOME/bin/mod_dav_svn.so on Windows
LoadModule authz_svn_module   modules/mod_authz_svn.so # Use full path to
SUBVERSION_HOME/bin/mod_authz_svn.so on Windows


# Work around authz and SVNListParentPath issue
RedirectMatch ^(/repos)$ $1/


# Enable Subversion logging
CustomLog logs/svn_logfile "%t %u %{SVN-ACTION}e" env=SVN-ACTION


<Location /repos/>
  # Enable Subversion
  DAV svn


  # Directory containing all repository for this path
  SVNParentPath /subversion/svn-repos


  # List repositories colleciton
```

```
  SVNListParentPath On

  # Enable WebDAV automatic versioning
  SVNAutoversioning On

  # Repository Display Name
  SVNReposName "Your Subversion Repository"

  # Do basic password authentication in the clear
  AuthType Basic

  # The name of the protected area or "realm"
  AuthName "Your Subversion Repository"

  # Make LDAP the authentication mechanism
  AuthBasicProvider ldap

  # Make LDAP authentication is final
  AuthzLDAPAuthoritative on

  # Active Directory requires an authenticating DN to access records
  AuthLDAPBindDN "CN=ldapuser,CN=Users,DC=your,DC=domain"

  # This is the password for the AuthLDAPBindDN user in Active Directory
  AuthLDAPBindPassword ldappassword

  # The LDAP query URL
  AuthLDAPURL "ldap://your.domain:389/DC=your,DC=domain?sAMAccountName?sub?
(objectClass=*)"

  # Require a valid user
  Require valid-user

  # Authorization file
  AuthzSVNAccessFile /subversion/apache2/auth/repos.acl
```

```
</Location>
```

Example Apache 2.0.x Configuration Snippet

```
# Load Apache LDAP modules
LoadModule ldap_module modules/mod_ldap.so
LoadModule auth_ldap_module modules/mod_auth_ldap.so


# Load Subversion Apache Modules
LoadModule dav_svn_module     modules/mod_dav_svn.so # Use full path to
SUBVERSION_HOME/bin/mod_dav_svn.so on Windows
LoadModule authz_svn_module   modules/mod_authz_svn.so # Use full path to
SUBVERSION_HOME/bin/mod_authz_svn.so on Windows


# Work around authz and SVNListParentPath issue
RedirectMatch ^(/repos)$ $1/


# Enable Subversion logging
CustomLog logs/svn_logfile "%t %u %{SVN-ACTION}e" env=SVN-ACTION


<Location /repos/>
  # Enable Subversion
  DAV svn


  # Directory containing all repository for this path
  SVNParentPath /subversion/svn-repos


  # List repositories colleciton
  SVNListParentPath On


  # Enable WebDAV automatic versioning
  SVNAutoversioning On


  # Repository Display Name
  SVNReposName "Your Subversion Repository"
```

```
  # LDAP Authentication is final
  AuthLDAPAuthoritative on

  # Do basic password authentication in the clear
  AuthType Basic

  # The name of the protected area or "realm"
  AuthName "Your Subversion Repository"

  # Active Directory requires an authenticating DN to access records
  AuthLDAPBindDN "CN=ldapuser,CN=Users,DC=your,DC=domain"

  # This is the password for the AuthLDAPBindDN user in Active Directory
  AuthLDAPBindPassword ldappassword

  # The LDAP query URL
  AuthLDAPURL "ldap://your.domain:389/DC=your,DC=domain?sAMAccountName?sub?
(objectClass=*)"

  # Require authentication
  Require valid-user

  # Authorization file
  AuthzSVNAccessFile /subversion/apache2/auth/repos.acl
</Location>
```

(The configurations above were for pointing to an Active Directory (AD) server.

Understanding the Configuration

So...the above Apache configurations are what I personally use when building an Apache-based server. Obviously there are changes that need to be made depending on the environment in but for now, it's a great start. To make the best of this opportunity, let's talk about the miscellaneous parts of the configuration.

SVNListParentPath and Subversion's authz

One of the first problems people run into when building an Apache-based Subversion server is when they want to have mod_dav_svn serve a list of repositories. Everything works fine until they enable Subversion's authorization (authz) support. What happens is the server will be configured

properly and secured properly but when you go to the repository collection list, which in our case is http://localhost/repos, you are forbidden to view the collection even if you have access. Well, with the **RedirectMatch** closer to the top of the configuration, you fix this issue. How you might be asking and the reason is that when you enable authz, you must have a trailing slash at the end of the collection url. With the RedirectMatch, we automatically redirect urls to the collection listing when there is no trailing slash. Problem solved.

Custom Subversion Logging

Subversion uses Apache's WebDAV support for providing access to its repositories when using Apache. Unfortunately, when you look at Apache's access logs to try and see your Subversion usage, you end up with a lot of WebDAV communication being logged and you only see a portion of the actual client/server communication. This is because mod_dav_svn uses Apache subrequests and Apache does not log subrequests. Even if it did, turning the Subversion communication in the Apache access log into something meaningful would be nearly impossible. That being said, the configuration above has been setup to use one of Subversion's features: Apache Logging which takes the guess work out.

Subversion Configuration

The other Subversion-specific parts of the Apache configuration are pretty self-explanitory. To summarize what is enabled with the above:

- SVNListParentPath: Enables the ability to browse the location root and get a list of repositories being served by that url base
- SVNAutoversioning: Enables the use of WebDAV clients to make changes to the repository contents without using a Subversion client
- SVNParentPath: Enables serving N number of repositories for the url base
- SVNReposName: Enables you to put in your own text to be visible in the web browser when browsing your repository contents via the built-in repository browser provided by mod_dav_svn
- AuthzSVNAccessFile: Tells Subversion's mod_authz_svn module where to find the authz file.

For more details about the Subversion-specific Apache directives, and a list of even more ways you can configure your Apache-based Subversion server, view the mod_dav_svn and the mod_authz_svn documentation.

LDAP Configuration

The LDAP portion of the Apache configuration is where most people run into problems. That being said, we'll spend a little more time explaining the Apache LDAP configuration. The most important thing to note is the subtle differences between Apache 2.0.x and Apache 2.2.x:

```
Apache 2.0.x           | Apache 2.2.x
-----------------------------------------------
AuthLDAPAuthoritative  | AuthzLDAPAuthoritative
AuthLDAPBindDN         | AuthLDAPBindDN
```

AuthLDAPBindPassword	AuthLDAPBindPassword
AuthLDAPURL	AuthLDAPURL
	AuthBasicProvider

You should note that the Apache LDAP module names have also changed between Apache 2.0.x and 2.2.x. Now that we see the naming changes, let's talk about how to properly use these Apache directives to get the LDAP-based authentication you're looking for. **(I will be using the Apache 2.2.x names for the Apache directives. If you're still using Apache 2.0.x, please refer to the table above for how to take my documentation and apply it to Apache 2.0.x.)**

- AuthzLDAPAuthoritative: Tells Apache whether or not a failed authentication request can be passed to other Apache modules
- AuthLDAPBindDN: The distinguished name of the user account that Apache will use to connect to the directory system to perform its user authentication
- AuthLDAPBindPassword: The password for the user account configured via the AuthLDAPBindDN directive
- AuthLDAPURL: This is a url that tells where the directory server is, where to look for users at, what user attribute is used to identify a user and other miscellaneous things specific to the LDAP query syntax (More on this later.)
- AuthBasicProvider: This tells Apache which authentication module you want to use for Basic authentication

All of the directives above are pretty straight forward except for the **AuthLDAPURL** directive. This directive we will discuss in more detail below. For any other Apache configuration questions, please resort to the Apache Documentation for your respective Apache version.

The LDAP Query URL

For most, the **AuthLDAPURL** directive is the most challenging to understand. There is good reason for this. That one directive actually consists of 6+ pieces of information that will be different for each Subversion server. Let's break our example **AuthLDAPURL** into its pieces and discuss the importance, and nuances, of each.

For simplicity, here is the url again, in its entirety: ldap://your.domain:389/DC=your,DC=domain?sAMAccountName?sub?(objectClass=*)

- Url scheme: [ldap] This is nothing more than a url scheme. It will usually be either 'ldap' or 'ldaps' in the event that you're using SSL for accessing your directory server.
- Hostname: [your.domain] This is the ip address or hostname of your directory server.
- Port: [389] This is the port the server is listening on for directory server communication.
- Search Base: [DC=your,DC=domain] This is the distinguished name to the path in the directory tree that you want to search for users.
- Username attribute: [sAMAccountName] This is the attribute contains the login name being used.
- Query scope: [sub] This tells the directory server what type of query to perform.
- Filter: [(objectClass=*)] This tells the directory server to filter the query for objects matching a particular filter

For more details on constructing an ldap url, which is a standard and not specific to Apache, view RFC 2255.

Working with Active Directory

Active Directory is known as a **Multi-Master Directory System**. This being said, each directory server in AD does not always have all the necessary information to perform all directory server requests. The best way to handle this is to have Apache query a **Global Catalog**. A Global Catalog server has the ability to search at the whole forest for users. This means if you want to do domain-wide searches or larger, you need to point to a Global Catalog and you need to update your Apache configuration accordingly. When using a Global Catalog, you should be using port 3268 when performing your queries.

Searching for Users

In the example url above, the **sAMAccountName** attribute is used to identify the username. This attribute is Windows/Active Directory specific so for those of you using OpenLDAP or another option, that attribute probably will not exist. Change your attribute accordingly. An example is if you wanted to use the **Common Name** to login, you could specify "CN" as the attribute.

LDAP Query Tuning

The last thing we will talk about is the ability to use filters to make your LDAP query a little more specific. In the example url above we used "(objectClass=*)", which will search for all objects. If you know that you only want to search for a particular object type, like the "user" type, you could use "(objectClass=user)" instead.

Conclusion

Building an Apache-based Subversion server with LDAP as the authentication mechanism can be daunting for some. I hope this has made things easier for you.