Authentication Plugins

1 Stackable Authentication Method(s)
- 1.1 Authentication by Password
  - 1.1.1 Enabling Authentication by Password
  - 1.1.2 Configuring Authentication by Password
- 1.2 Shibboleth Authentication
  - 1.2.1 Enabling Shibboleth Authentication
  - 1.2.2 Configuring Shibboleth Authentication
    - 1.2.2.1 Apache "mod_shib" Configuration (required)
    - 1.2.2.2 Sample shibboleth2.xml Configuration
    - 1.2.2.3 DSpace Shibboleth Configuration Options
- 1.3 LDAP Authentication
  - 1.3.1 Introduction to LDAP specific terminology
  - 1.3.2 Enabling LDAP Authentication
  - 1.3.3 Configuring LDAP Authentication
  - 1.3.4 Debugging LDAP connection and configuration
  - 1.3.5 Enabling Hierarchical LDAP Authentication
  - 1.3.6 Configuring Hierarchical LDAP Authentication
- 1.4 IP Authentication
  - 1.4.1 Enabling IP Authentication
  - 1.4.2 Configuring IP Authentication
- 1.5 X.509 Certificate Authentication
  - 1.5.1 Enabling X.509 Certificate Authentication
  - 1.5.2 Configuring X.509 Certificate Authentication
- 1.6 Example of a Custom Authentication Method

Stackable Authentication Method(s)

Since many institutions and organizations have existing authentication systems, DSpace has been designed to allow these to be easily integrated into an existing authentication infrastructure. It keeps a series, or "stack", of *authentication methods*, so each one can be tried in turn. This makes it easy to add new authentication methods or rearrange the order without changing any existing code. You can also share authentication code with other sites.

Configuration File:	**`[dspace]/config/modules/authentication.cfg`**
Property:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod`
Example Value:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.PasswordAuthentication`

The configuration property `plugin.sequence.org.dspace.authenticate.AuthenticationMethod` defines the authentication stack. It is a comma-separated list of class names. Each of these classes implements a different `authentication method`, or way of determining the identity of the user. They are invoked in the order specified until one succeeds.

Existing Authentication Methods include

- Authentication by Password (class: `org.dspace.authenticate.PasswordAuthentication`) (DEFAULT)
- Shibboleth Authentication (class: `org.dspace.authenticate.ShibAuthentication`)
- LDAP Authentication (class: `org.dspace.authenticate.LDAPAuthentication`)
- IP Address based Authentication (class: `org.dspace.authenticate.IPAuthentication`)
- X.509 Certificate Authentication (class: `org.dspace.authenticate.X509Authentication`)

An authentication method is a class that implements the interface `org.dspace.authenticate.AuthenticationMethod`. It `authenticates` a user by evaluating the *credentials* (e.g. username and password) he or she presents and checking that they are valid.

The basic authentication procedure in the DSpace Web UI is this:

1. A request is received from an end-user's browser that, if fulfilled, would lead to an action requiring authorization taking place.
2. If the end-user is already authenticated:
   - If the end-user is allowed to perform the action, the action proceeds
   - If the end-user is NOT allowed to perform the action, an authorization error is displayed.
   - If the end-user is NOT authenticated, i.e. is accessing DSpace anonymously:
3. The parameters etc. of the request are stored.
4. The Web UI's `startAuthentication` method is invoked.
5. First it tries all the authentication methods which do `implicit` authentication (i.e. they work with just the information already in the Web request, such as an X.509 client certificate). If one of these succeeds, it proceeds from Step 2 above.
6. If none of the implicit methods succeed, the UI responds by putting up a "login" page to collect credentials for one of the `explicit` authentication methods in the stack. The servlet processing that page then gives the proffered credentials to each authentication method in turn until one succeeds, at which point it retries the original operation from Step 2 above.
   Please see the source files `AuthenticationManager.java` and `AuthenticationMethod.java` for more details about this mechanism.

Authentication by Password

Enabling Authentication by Password

By default, this authentication method is enabled in DSpace.

However, to enable Authentication by Password, you must ensure the `org.dspace.authenticate.PasswordAuthentication` class is listed as one of the AuthenticationMethods in the following configuration:

Configuration File:	**[dspace]/config/modules/authentication.cfg**
Property:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod`
Example Value:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.PasswordAuthentication`

Configuring Authentication by Password

The default method `org.dspace.authenticate.PasswordAuthentication` has the following properties:

- Use of built-in e-mail address/password-based log-in. This is achieved by forwarding a request that is attempting an action requiring authorization to the password log-in servlet, `/password-login`. The password log-in servlet (`org.dspace.app.webui.servlet.PasswordServlet`) contains code that will resume the original request if authentication is successful, as per step 3. described above.
- Users can register themselves (i.e. add themselves as e-people without needing approval from the administrators), and can set their own passwords when they do this
- Users are not members of any special (dynamic) e-person groups
- You can restrict the domains from which new users are able to register. To enable this feature, uncomment the following line from dspace.cfg: `authentication.password.domain.valid = example.com` Example options might be '`@example.com`' to restrict registration to users with addresses ending in @example.com, or '`@example.com, .ac.uk`' to restrict registration to users with addresses ending in @example.com or with addresses in the .ac.uk domain.

A full list of all available Password Authentication Configurations:

Configuration File:	**[dspace]/config/modules/authentication-password.cfg**
Property:	`authentication-password.domain.valid`
Example Value:	`authentication-password.domain.value = @mit.edu, .ac.uk`
Informational Note:	This option allows you to limit self-registration to email addresses ending in a particular domain value. The above example would limit self-registration to individuals with "@mit.edu" email addresses and all ".ac.uk" email addresses.
Property:	`authentication-password.login.specialgroup`
Example Value:	`authentication-password.login.specialgroup = My DSpace Group`
Informational Note:	This option allows you to automatically add all password authenticated user sessions to a specific DSpace Group (the group must exist in DSpace) for the remainder of their logged in session.
Property:	`authentication-password.digestAlgorithm`
Example Value:	`authentication-password.digestAlgorithm = SHA-512`
Informational Note:	This option specifies the hashing algorithm to be used in converting plain-text passwords to more secure password digests. The example value is the default. You may select any digest algorithm available through java.security.MessageDigest on your system. At least MD2, MD5, SHA-1, SHA-256, SHA-384, and SHA-512 should be available, but you may have installed others. Most sites will not need to adjust this.

Shibboleth Authentication

Enabling Shibboleth Authentication

To enable Shibboleth Authentication, you must ensure the `org.dspace.authenticate.ShibAuthentication` class is listed as one of the AuthenticationMethods in the following configuration:

Configuration File:	**[dspace]/config/modules/authentication.cfg**

Property:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod`
Example Value:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.ShibAuthentication`

Configuring Shibboleth Authentication

Shibboleth is a distributed authentication system for securely authenticating users and passing attributes about the user from one or more identity providers. In the Shibboleth terminology DSpace is a Service Provider which receives authentication information and then based upon that provides a service to the user. To use Shibboleth, DSpace *requires* that you use Apache installed with the mod_shib module acting as a proxy for all HTTP requests for your servlet container (typically Tomcat). DSpace will receive authentication information from the mod_shib module through HTTP headers.

Before DSpace will work with Shibboleth, you **must** have the following:

1. An Apache web server with the "mod_shib" module installed. As mentioned, this mod_shib module acts as a proxy for all HTTP requests for your servlet container (typically Tomcat). Any requests to DSpace that require authentication via Shibboleth should be redirected to 'shibd' (the shibboleth daemon) by this "mod_shib" module. Details on installing/configuring mod_shib in Apache are available at: https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig We also have a sample Apache + mod_shib configuration provided below.
2. An external Shibboleth IdP (Identity Provider). Using mod_shib, DSpace will only act as a Shibboleth SP (Service Provider). The actual Shibboleth Authentication & Identity information must be provided by an external IdP. If you are using Shibboleth at your institution already, then there already should be a Shibboleth IdP available. More information about Shibboleth IdPs versus SPs is available at: https://wiki.shibboleth.net/confluence/display/SHIB2/UnderstandingShibboleth

For more information on installing and configuring a Shibboleth Service Provider see: https://wiki.shibboleth.net/confluence/display/SHIB2/Installation

Note about Shibboleth Active vs Lazy Sessions:

When configuring your Shibboleth Service Provider there are two Shibboleth paradigms you may use: Active or Lazy Sessions. Active sessions is where the mod_shib module is configured to product an entire URL space. No one will be able to access that URL without first authenticating with Shibboleth. Using this method you will need to configure shibboleth to protect the URL: "/shibboleth-login". The alternative, Lazy Session does not protect any specific URL. Instead Apache will allow access to any URL, and when the application wants to it may initiate an authenticated session.

The Lazy Session method is preferable for most DSpace installations, as you usually want to provide public access to (most) DSpace content, while restricting access to only particular areas (e.g. administration UI/tools, private Items, etc.). When Active Sessions are enabled your *entire* DSpace site will be access restricted. In other words, when using Active Sessions, Shibboleth will require everyone to first authenticate before they can access any part of your repository (which essentially results in a "dark archive", as anonymous access will not be allowed).

Apache "mod_shib" Configuration (required)

As mentioned above, you must have Apache with the "mod_shib" module installed in order for DSpace to be able to act as a Shibboleth Service Provider (SP). The mod_shib module acts as a proxy for all HTTP requests for your servlet container (typically Tomcat). Any requests to DSpace that require authentication via Shibboleth should be redirected to 'shibd' (the shibboleth daemon) by this "mod_shib" module. Details on installing/configuring mod_shib in Apache are available at: https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig General information about installing /configuring Shibboleth Service Providers (SPs) can be found at: https://wiki.shibboleth.net/confluence/display/SHIB2/Installation

A few extra notes/hints when configuring mod_shib & Apache:

- The Shibboleth setting "ShibUseHeaders" is no longer required to be set to "On", as DSpace will correctly utilize attributes instead of headers.
    - When "ShibUseHeaders" is set to "Off" (which is recommended in the mod_shib documentation), proper configuration of Apache to pass attributes to Tomcat (via either mod_jk or mod_proxy) can be a bit tricky, SWITCH has some great documentation on exactly what you need to do. We will eventually paraphrase/summarize this documentation here, but for now, the SWITCH page will have to do.
- When initially setting up Apache & mod_shib, https://www.testshib.org/ provides a great testing ground for your configurations. This site provides a sample/demo Shibboleth IdP (as well as a sample Shibboleth SP) which you can test against. It acts as a "sandbox" to get your configurations working properly, before you point DSpace at your production Shibboleth IdP.

Below, we have provided a sample Apache configuration. However, as every institution has their own specific Apache setup/configuration, it is highly likely that you will need to tweak this configuration in order to get it working properly. Again, see the official mod_shib documentation for much more detail about each of these settings: https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig These configurations are meant to be added to an Apache <VirtualHost> which acts as a proxy to your Tomcat (or other servlet container) running DSpace. More information on Apache VirtualHost settings can be found at: https://httpd.apache.org/docs/2.2/vhosts/

```
#### SAMPLE MOD_SHIB CONFIGURATION FOR APACHE2 (it may require local modifications based on your Apache setup)
####
# While this sample VirtualHost is for HTTPS requests (recommended for Shibboleth, obviously),
# you may also need/want to create one for HTTP (*:80)
<VirtualHost *:443>
   ...
   # PLEASE NOTE: We have omitted many Apache settings (ServerName, LogLevel, SSLCertificateFile, etc)
   # which you may need/want to add to your VirtualHost

   # As long as Shibboleth module is installed, enable all Shibboleth/mod_shib related settings
   <IfModule mod_shib>
       # Shibboleth recommends turning on UseCanonicalName
       # See "Prepping Apache" in https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig
       UseCanonicalName On

       # Most DSpace instances will want to use Shibboleth "Lazy Session", which ensures that users
       # can access DSpace without first authenticating via Shibboleth.
       # This section turns on Shibboleth "Lazy Session". Also ensures that once they have authenticated
       # (by accessing /Shibboleth.sso/Login path), then their Shib session is kept alive
       <Location />
         AuthType shibboleth
         ShibRequireSession Off
         require shibboleth
         # If your "shibboleth2.xml" file specifies an <ApplicationOverride> setting for your
         # DSpace Service Provider, then you may need to tell Apache which "id" to redirect Shib requests to.
         # Just uncomment this and change the value "my-dspace-id" to the associated @id attribute value.
         #ShibRequestSetting applicationId my-dspace-id
       </Location>

       # If a user attempts to access the DSpace shibboleth login page, force them to authenticate via Shib
       <Location "/shibboleth-login">
         AuthType shibboleth
         ShibRequireSession On
         # Please note that setting ShibUseHeaders to "On" is a potential security risk.
         # You may wish to set it to "Off". See the mod_shib docs for details about this setting:
         # https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig#NativeSPApacheConfig-
AuthConfigOptions
         # Here's a good guide to configuring Apache + Tomcat when this setting is "Off":
         # https://www.switch.ch/de/aai/support/serviceproviders/sp-access-rules.html#javaapplications
         ShibUseHeaders On
         require valid-user
       </Location>

       # Ensure /Shibboleth.sso path (in Apache) can be accessed
       # By default it may be inaccessible if your Apache security is tight.
       <Location "/Shibboleth.sso">
         Order deny,allow
         Allow from all
         # Also ensure Shibboleth/mod_shib responds to this path
         SetHandler shib
       </Location>

       # Finally, you may need to ensure requests to /Shibboleth.sso are NOT redirected
       # to Tomcat (as they need to be handled by mod_shib instead).
       # NOTE: THIS SETTING IS LIKELY ONLY NEEDED IF YOU ARE USING mod_proxy TO REDIRECT
       # ALL REQUESTS TO TOMCAT (e.g. ProxyPass / ajp://localhost:8009/)
       # ProxyPass /Shibboleth.sso !
   </IfModule>

   ...

</VirtualHost>
```

Sample shibboleth2.xml Configuration

In addition, here's a sample "ApplicationOverride" configuration for "shibboleth2.xml". This particular "ApplicationOverride" is configured to use the Test IdP provided by http://www.testshib.org/ and is just meant as an example. In order to enable it for testing purposes, you **must** specify `ShibRequestSetting applicationId testshib` in your Apach `mod_shib` configuration (see above).

```
        <!-- *** Sample Shibboleth Settings for http://www.testshib.org/ ***     -->
        <!-- This provides a simple sample of how you could configure            -->
        <!-- shibboleth2.xml for DSpace sites.                                   -->
        <!-- TO ENABLE: You'd need to specify "applicationId" as "testshib" in   -->
        <!-- your mod_shib settings, e.g.                                        -->
        <!-- <Location />                                                        -->
        <!--     ...                                                             -->
        <!--     ShibRequestSetting applicationId testshib                       -->
        <!-- </Location>                                                         -->
        <ApplicationOverride id="testshib" entityID="http://mydspace.edu/shibboleth" REMOTE_USER="principal-id">

            <!-- We'll use a TEST IdP, hosted by the awesome http://www.testshib.org/ testing service. -->
            <!-- See also: https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPServiceSSO -->
            <Sessions lifetime="28800" timeout="3600" checkAddress="false" relayState="ss:mem" handlerSSL="true"
>
               <SSO entityID="https://idp.testshib.org/idp/shibboleth">
                 SAML2 SAML1
               </SSO>
            </Sessions>

            <!-- Loads and trusts a metadata file that describes the IdP and how to communicate with it. -->
            <!-- By default, metadata is retrieved from the TEST IdP at http://www.testshib.org -->
            <!-- and is cached in a local file named "testshib-idp-metadata.xml". -->
            <!-- See also: https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPMetadataProvider -->
            <MetadataProvider type="XML" uri="http://www.testshib.org/metadata/testshib-providers.xml"
                          backingFilePath="testshib-idp-metadata.xml" reloadInterval="180000"/>
        </ApplicationOverride>
```

DSpace Shibboleth Configuration Options

Authentication Methods:

DSpace supports authentication using NetID, or email address. A user's NetID is a unique identifier from the IdP that identifies a particular user. The NetID can be of almost any form such as a unique integer, string, or with Shibboleth 2.0 you can use "targeted ids". You will need to coordinate with your shibboleth federation or identity provider. There are three ways to supply identity information to DSpace:

*1) NetID from Shibboleth Header (**best**)*

The NetID-based method is superior because users may change their email address with the identity provider. When this happens DSpace will not be able to associate their new address with their old account.

*2) Email address from Shibboleth Header (**okay**)*

In the case where a NetID header is not available or not found DSpace will fall back to identifying a user based-upon their email address.

*3) Tomcat's Remote User (**worst**)*

In the event that neither Shibboleth headers are found then as a last resort DSpace will look at Tomcat's remote user field. This is the least attractive option because Tomcat has no way to supply additional attributes about a user. Because of this the autoregister option is not supported if this method is used.

Identity Scheme Migration Strategies:

If you are currently using Email based authentication (either 1 or 2) and want to upgrade to NetID based authentication then there is an easy path. Simply enable shibboleth to pass the NetID attribute and set the netid-header below to the correct value. When a user attempts to log in to DSpace first DSpace will look for an EPerson with the passed NetID, however when this fails DSpace will fall back to email based authentication. Then DSpace will update the user's EPerson account record to set their NetID so all future authentications for this user will be based upon NetID. One thing to note is that DSpace will prevent an account from switching NetIDs. If an account already has a NetID set and then they try and authenticate with a different NetID the authentication will fail.

EPerson Metadata:

One of the primary benefits of using Shibboleth based authentication is receiving additional attributes about users such as their names, telephone numbers, and possibly their academic department or graduation semester if desired. DSpace treats the first and last name attributes differently because they (along with email address) are the three pieces of minimal information required to create a new user account. For both first and last name supply direct mappings to the Shibboleth headers. In additional to the first and last name DSpace supports other metadata fields such as phone, or really anything you want to store on an eperson object. Beyond the phone field, which is accessible in the user's profile screen, none of these additional metadata fields will be used by DSpace out-of-the box. However if you develop any local modification you may access these attributes from the EPerson object. The Vireo ETD workflow system utilizes this to aid students when submitting an ETD.

Role-based Groups:

DSpace is able to place users into pre-defined groups based upon values received from Shibboleth. Using this option you can place all faculty members into a DSpace group when the correct affiliation's attribute is provided. When DSpace does this they are considered 'special groups', these are really groups but the user's membership within these groups is not recorded in the database. Each time a user authenticates they are automatically placed within the pre-defined DSpace group, so if the user loses their affiliation then the next time they login they will no longer be in the group.

Depending upon the shibboleth attributed use in the role-header it may be scoped. Scoped is shibboleth terminology for identifying where an attribute originated from. For example a students affiliation may be encoded as "student@tamu.edu". The part after the @ sign is the scope, and the preceding value is the value. You may use the whole value or only the value or scope. Using this you could generate a role for students and one institution different than students at another institution. Or if you turn on ignore-scope you could ignore the institution and place all students into one group.

The values extracted (a user may have multiple roles) will be used to look up which groups to place the user into. The groups are defined as "authentication -shibboleth.role.<role-name>" which is a comma separated list of DSpace groups.

Configuration File:	**`[dspace]/config/modules/authentication-shibboleth.cfg`**
Property:	`authentication-shibboleth.lazysession`
Example Value:	`authentication-shibboleth.lazysession = true`
Informational Note:	Whether to use lazy sessions or active sessions. For more DSpace instances, you will likely want to use lazy sessions. Active sessions will force **every** user to authenticate via Shibboleth before they can access your DSpace (essentially resulting in a "dark archive").
Property:	`authentication-shibboleth.lazysession.loginurl`
Example Value:	authentication-shibboleth.`lazysession.loginurl = /Shibboleth.sso/Login`
Informational Note:	The url to start a shibboleth session (only for lazy sessions). Generally this setting will be "/Shibboleth.sso/Login"
Property:	`authentication-shibboleth.lazysession.secure`
Example Value:	`authentication-shibboleth.lazysession.secure = true`
Informational Note:	Force HTTPS when authenticating (only for lazy sessions). Generally this is recommended to be "true".
Property:	`authentication-shibboleth.netid-header`
Example Value:	`authentication-shibboleth.netid-header = SHIB-NETID`
Informational Note:	The HTTP header where shibboleth will supply a user's NetID. This HTTP header should be specified as an Attribute within your Shibboleth "attribute-map.xml" configuration file.
Property:	`authentication-shibboleth.email-header`
Example Value:	`authentication-shibboleth.email-header = SHIB-MAIL`
Informational Note:	The HTTP header where the shibboleth will supply a user's email address. This HTTP header should be specified as an Attribute within your Shibboleth "attribute-map.xml" configuration file.
Property:	`authentication-shibboleth.email-use-tomcat-remote-user`
Example Value:	`authentication-shibboleth.email-use-tomcat-remote-user = false`
Informational Note:	Used when a netid or email headers are not available should Shibboleth authentication fall back to using Tomcat's remote user feature? Generally this is not recommended. See the "Authentication Methods" section above.
Property:	`authentication-shibboleth.reconvert.attributes`
Example Value	`authentication-shibboleth.reconvert.attributes = false`

Informational Note:	Shibboleth attributes are by default UTF-8 encoded. Some servlet container automatically converts the attributes from ISO-8859-1 (latin-1) to UTF-8. As the attributes already were UTF-8 encoded it may be necessary to reconvert them. If you set this property true, DSpace converts all shibboleth attributes retrieved from the servlet container from UTF-8 to ISO-8859-1 and uses the result as if it were UTF-8. This procedure restores the shibboleth attributes if the servlet container wrongly converted them from ISO-8859-1 to UTF-8. Set this true, if you notice character encoding problems within shibboleth attributes.
Property:	`authentication-shibboleth.autoregister`
Example Value:	`authentication-shibboleth.autoregister = true`
Informational Note:	Should we allow new users to be registered automatically?
Property:	`authentication-shibboleth.sword.compatibility`
Example Value:	`authentication-shibboleth.sword.compatibility = false`
Informational Note:	SWORD compatibility will allow this authentication method to work when using SWORD. SWORD relies on username and password based authentication and is entirely incapable of supporting shibboleth. This option allows you to authenticate username and passwords for SWORD sessions with out adding another authentication method onto the stack. You will need to ensure that a user has a password. One way to do that is to create the user via the create-administrator command line command and then edit their permissions. WARNING: If you enable this option while ALSO having "PasswordAuthentication" enabled, then you should ensure that "PasswordAuthentication" is listed prior to "ShibAuthentication" in your authentication.cfg file. Otherwise, ShibAuthentication will be used to authenticate all of your users INSTEAD OF PasswordAuthentication.
Property:	`authentication-shibboleth.firstname-header`
Example Value:	`authentication-shibboleth.firstname-header = SHIB-GIVENNAME`
Informational Note:	The HTTP header where the shibboleth will supply a user's given name. This HTTP header should be specified as an Attribute within your Shibboleth "attribute-map.xml" configuration file.
Property:	`authentication-shibboleth.lastname-header`
Example Value:	`authentication-shibboleth.lastname-header = SHIB-SN`
Informational Note:	The HTTP header where the shibboleth will supply a user's surname. This HTTP header should be specified as an Attribute within your Shibboleth "attribute-map.xml" configuration file.
Property:	`authentication-shibboleth.eperson.metadata`

Example Value:	`authentication-shibboleth.eperson.metadata = \`
	` SHIB-telephone => phone, \`
	` SHIB-cn => cn`

Informational Note:	Additional user attributes mapping, multiple attributes may be stored for each user. The left side is the Shibboleth-based metadata Header and the right side is the eperson metadata field to map the attribute to.
Property:	`authentication-shibboleth.eperson.metadata.autocreate`
Example Value:	`authentication-shibboleth.eperson.metadata.autocreate = true`
Informational Note:	If the eperson metadata field is not found, should it be automatically created?
Property:	`authentication-shibboleth.role-header`
Example Value:	`authentication-shibboleth.role-header = SHIB-SCOPED-AFFILIATION`
Informational Note:	The shibboleth header to do role-based mappings. This HTTP header should be specified as an Attribute within your Shibboleth "attribute-map.xml" configuration file. See the **Role-based Groups** section above for more info.
Property:	`authentication-shibboleth.role-header.ignore-scope`
Example Value:	`authentication-shibboleth.role-header.ignore-scope = true`
Informational Note:	Weather to ignore the attribute's scope (everything after the @ sign for scoped attributes)
Property:	`authentication-shibboleth.role-header.ignore-value`
Example Value:	`authentication-shibboleth.role-header.ignore-value = false`

Informational Note:	Weather to ignore the attribute's value (everything before the @ sign for scoped attributes)
Property:	`authentication-shibboleth.role.[affiliation-attribute]`

Example Value:	`authentication-shibboleth.role.faculty = Faculty, Member`
	`authentication-shibboleth.role.staff = Staff, Member`
	`authentication-shibboleth.role.student = Students, Member`

Informational Note:	Mapping of affiliation values to DSpace groups. See the **Role-based Groups** section above for more info.

LDAP Authentication

Introduction to LDAP specific terminology

If you are unfamiliar with LDAP, the following introduction to some of its terminology might come in handy:

https://stackoverflow.com/questions/18756688/what-are-cn-ou-dc-in-an-ldap-search

Enabling LDAP Authentication

To enable LDAP Authentication, you must ensure the `org.dspace.authenticate.LDAPAuthentication` class is listed as one of the AuthenticationMethods in the following configuration:

Configuration File:	**`[dspace]/config/modules/authentication.cfg`**
Property:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod`
Example Value:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.LDAPAuthentication`

Configuring LDAP Authentication

If LDAP is enabled, then new users will be able to register by entering their username and password without being sent the registration token. If users do not have a username and password, then they can still register and login with just their email address the same way they do now.

If you want to give any special privileges to LDAP users, create a stackable authentication method to automatically put people who have a netid into a special group. You might also want to give certain email addresses special privileges. Refer to the Custom Authentication Code section below for more information about how to do this.

Ensure required commas are escaped in LDAP configuration

NOTE: As of DSpace 6, commas (,) are now a special character in the Configuration system. As some LDAP configuration may contain commas, you must be careful to escape any required commas by adding a backslash (\) before each comma, e.g. "\,". The configuration reference for authentication-ldap.cfg has been updated below with additional examples.

Here is an explanation of each of the different LDAP configuration parameters:

Configuration File:	**`[dspace]/config/modules/authentication-ldap.cfg`**
Property:	`authentication-ldap.enable`
Example Value:	`authentication-ldap.enable = false`
Informational Note:	This setting will enable or disable LDAP authentication in DSpace. With the setting off, users will be required to register and login with their email address. With this setting on, users will be able to login and register with their LDAP user ids and passwords.
Property:	`authentication-ldap.autoregister`
Example Value:	`authentication-ldap.autoregister = true`

Informational Note:	This will turn LDAP autoregistration on or off. With this on, a new EPerson object will be created for any user who successfully authenticates against the LDAP server when they first login. With this setting off, the user must first register to get an EPerson object by entering their ldap username and password and filling out the forms.
Property:	`authentication-ldap.provider_url`
Example Value:	`authentication-ldap.provider_url = ldap://ldap.myu.edu/o=myu.edu\,ou=mydept`
Informational Note:	This is the url to your institution's LDAP server. You may or may not need the /o=myu.edu part at the end. Your server may also require the ldaps:// protocol. (This field has no default value) **NOTE:** As of DSpace 6, commas (,) are now a special character in the Configuration system. Therefore, be careful to escape any required commas in this configuration by adding a backslash (\) before each comma, e.g. "\,"
Property:	`authentication-ldap.starttls`
Example Value:	`authentication-ldap.starttls = false`
Informational Note:	Should we issue StartTLS after establishing TCP connection in order to initiate an encrypted connection? Note: This (TLS) is different from LDAPS: • TLS is a tunnel for plain LDAP and is typically recognized on the same port (standard LDAP port: 389) • LDAPS is a separate protocol, deprecated in favor of the standard TLS method. (standard LDAPS port: 636)
Property:	`authentication-ldap.id_field`
Example Value:	`authentication-ldap.id_field = uid`
Explanation:	This is the unique identifier field in the LDAP directory where the username is stored. (This field has no default value)
Property:	`authentication-ldap.object_context`
Example Value:	`authentication-ldap.object_context = ou=people\,o=myu.edu`
Informational Note:	This is the LDAP object context to use when authenticating the user. By default, DSpace will use this value to create the user's DN in order to attempt to authenticate them. It is appended to the *id_field* and username. For example `uid=username\,ou=people\,o=myu.edu`. You will need to modify this to match your LDAP configuration. (This field has no default value) If your users do NOT all exist under a single "object_context" in LDAP, then you should ignore this setting and INSTEAD use the Hierarchical LDAP Authentication settings below (especially see "`search.user`" or "`search.anonymous`") **NOTE:** As of DSpace 6, commas (,) are now a special character in the Configuration system. Therefore, be careful to escape any required commas in this configuration by adding a backslash (\) before each comma, e.g. "\,"
Property:	`authentication-ldap.search_context`
Example Value:	`authentication-ldap.search_context = ou=people`
Informational Note:	This is the search context used when looking up a user's LDAP object to retrieve their data for autoregistering. With `autoregister=true`, when a user authenticates without an EPerson object we search the LDAP directory to get their name (`id_field`) and email address (`email_field`) so that we can create one for them. So after we have authenticated against *uid=username, ou=people,o=byu.edu* we now search in ou=people for filtering on [uid=username]. Often the `search_context` is the same as the `object_context` parameter. But again this depends on your LDAP server configuration. (This field has no default value, and it MUST be specified when either `search.anonymous=true` or `search.user` is specified) **NOTE:** As of DSpace 6, commas (,) are now a special character in the Configuration system. Therefore, be careful to escape any required commas in this configuration by adding a backslash (\) before each comma, e.g. "\,"
Property:	`authentication-ldap.email_field`
Example Value:	`authentication-ldap.email_field = mail`
Informational Note:	This is the LDAP object field where the user's email address is stored. "mail" is the most common for LDAP servers. (This field has no default value) If the "email_field" is unspecified, or the user has no email address in LDAP, his/her username (id_field value) will be saved as the email in DSpace (or appended to `netid_email_domain`, when specified)
Property:	`authentication-ldap.netid_email_domain`
Example Value:	`authentication-ldap.netid_email_domain = @example.com`

Informational Note:	If your LDAP server does not hold an email address for a user (i.e. no `email_field`), you can use the following field to specify your email domain. This value is appended to the netid (`id_field`) in order to make an email address (which is then stored in the DSpace EPerson). For example, a netid of 'user' and `netid_email_domain` as `@example.com` would set the email of the user to be user@example.com *Please note:* this field will only be used if "`email_field`" is unspecified OR the user in question has no email address stored in LDAP. If both "`email_field`" and "`netid_email_domain`" are unspecified, then the "`id_field`" will be used as the email address.
Property:	`authentication-ldap.surname_field`
Example Value:	`authentication-ldap.surname_field = sn`
Informational Note:	This is the LDAP object field where the user's last name is stored. "`sn`" is the most common for LDAP servers. If the field is not found the field will be left blank in the new eperson object. (This field has no default value)
Property:	`authentication-ldap.givenname_field`
Example Value:	`authentication-ldap.givenname_field = givenName`
Informational Note:	This is the LDAP object field where the user's given names are stored. I'm not sure how common the givenName field is in different LDAP instances. If the field is not found the field will be left blank in the new eperson object. (This field has no default value)
Property:	`authentication-ldap.phone_field`
Example Value:	`authentication-ldap.phone_field = telephoneNumber`
Informational Note:	This is the field where the user's phone number is stored in the LDAP directory. If the field is not found the field will be left blank in the new eperson object. (This field has no default value)
Property:	`authentication-ldap.login.specialgroup`
Example Value:	`authentication-ldap.login.specialgroup = group-name`
Informational Note:	If specified, all user sessions successfully logged in via LDAP will automatically become members of this DSpace Group (for the remainder of their current, logged in session). This DSpace Group **must** already exist (it will not be automatically created). This is useful if you want a DSpace Group made up of all internal authenticated users. This DSpace Group can then be used to bestow special permissions on any users who have authenticated via LDAP (e.g. you could allow anyone authenticated via LDAP to view special, on campus only collections or similar)
Property:	`authentication-ldap.login.groupmap.*`

Example Value:	`authentication-ldap.login.groupmap.1 = ou=Students:ALL_STUDENTS`
	`authentication-ldap.login.groupmap.2 = ou=Employees:ALL_EMPLOYEES`
	`authentication-ldap.login.groupmap.3 = ou=Faculty:ALL_FACULTY`

Informational Note:	The left part of the value (before the ":") must correspond to a portion of a user's DN (unless "`login.group.attribute`" is specified..please see below). The right part of the value corresponds to the name of an existing DSpace group.
	
	For example, if the authenticated user's DN in LDAP is in the following form:
	
	`cn=jdoe,OU=Students,OU=Users,dc=example,dc=edu`
	
	that user would get assigned to the ALL_STUDENTS DSpace group for the remainder of their current session.
	
	However, if that same user later graduates and is employed by the university, their DN in LDAP may change to:
	
	`cn=jdoe,OU=Employees,OU=Users,dc=example,dc=edu`
	
	Upon logging into DSpace after that DN change, the authenticated user would now be assigned to the ALL_EMPLOYEES DSpace group for the remainder of their current session.
	
	Note: This option can be used independently from the login.specialgroup option, which will put all LDAP users into a single DSpace group. Both options may be used together.

Property:	`authentication-ldap.login.groupmap.attribute`
Example Value:	`authentication-ldap.login.groupmap.attribute = group`

Informational Note:	The value of the "`authentication-ldap.login.groupmap.attribute`" should specify the name of a single LDAP attribute. If this property is uncommented, it changes the meaning of the left part of "`authentication-ldap.login.groupmap.*`" (see above) as follows: • If the authenticated user has this LDAP attribute, look up the value of this LDAP attribute in the left part (before the ":") of the `authentication-ldap.login.groupmap.*` value • If that LDAP value is found in any "`authentication-ldap.login.groupmap.*`" field, assign this authenticated user to the DSpace Group specified by the right part (after the ":") of the `authentication-ldap.login.groupmap.*` value. For example: • `authentication-ldap.login.groupmap.attribute = group` • `authentication-ldap.login.groupmap.1 = mathematics:Mathematics_Group` The above would ensure that any authenticated users where their LDAP "group" attribute equals "mathematics" would be added to the DSpace Group named "Mathematics_Group" for the remainder of their current session. However, if that same user logged in later with a new LDAP "group" value of "computer science", he/she would no longer be a member of the "Mathematics_Group" in DSpace.

Debugging LDAP connection and configuration

As every LDAP is different, configuring your DSpace to communicate with your LDAP can sometimes be a challenge. We recommend using third-party LDAP tools to test your LDAP connection / username / password, and perform sample searches to better understand what information is being returned from your local LDAP. This will help ensure that LDAP configuration goes more smoothly.

One example of such an LDAP tool is the `ldapsearch` commandline tool available in most Linux operating systems (e.g. in Debian / Ubuntu it's available in the "ldap-utils" package). Below are some example ldapsearch commands that can be used to determine (and/or debug) specific configurations in your `authentication-ldap.cfg`. In the below examples, we've used the names of specific DSpace configurations as placeholders (in square brackets).

```
# Basic anonymous connection (for VERBOSE, add -v)
ldapsearch -x -H [provider_url]

# Debug a connection error (add -d-1)
# If you are connecting to an LDAPS URL and see connection errors (e.g. "peer cert untrusted or revoked")
# then see below note about "SSL Connection Errors"
ldapsearch -x -H [provider_url] -d-1


# Attempt to connect to [provider_url] as [search.user] (will prompt for search.user's password)
# This doesn't actually perform a query, just ensures that authentication is working
# NOTE: "search.user" is USUALLY either the full user DN (e.g. "cn=dspaceadmin,ou=people,o=myu.edu")
# or "DOMAIN\USERNAME" (e.g. "MYU\DSpaceUser"). The latter is more likely with Windows Active Directory
ldapsearch -x -H [provider_url] -D [search.user] -W


# Attempt to list the first 100 users in a given [search_context], returning the "cn", "mail" and "sn" fields
for each
ldapsearch -x -H [provider_url] -D [search.user] -W -b [search_context] -z 100 cn mail sn


# Attempt to find the first 100 users whose [id_field] starts with the letter "t", returning the [id_field],
"cn", "mail" and "sn" fields for each
ldapsearch -x -H [provider_url] -D [search.user] -W -b [search_context] -z 100 -s sub "([id_field]=t*)"
[id_field] cn mail sn
```

SSL Connection Errors: If you are using ldapsearch with an LDAPS connection (secure connection), you may receive "peer cert untrusted or revoked" errors if the LDAP SSL certificate is self-signed. You can temporarily tell LDAP to accept any security certificate by setting `TLS_REQCERT allow` in your ldapsearch's `ldap.conf` file. *Be sure to remove this setting however after you are done testing!*

```
# FOR TESTING ONLY! This setting disables the check for a valid LDAP Server security certificate,
# which is considered a security issue for production LDAP setups. Setting this to "allow" tells
# the LDAP client to accept any security certificates that it cannot verify or validate.
TLS_REQCERT allow
```

More information on this SSL workaround can be found at:

- http://www.bind9.net/manual/openldap/2.3/tls.html
- http://muzso.hu/2012/03/29/how-to-configure-ssl-aka.-ldaps-for-libnss-ldap-auth-client-config-in-ubuntu

Enabling Hierarchical LDAP Authentication

Please note, that DSpace doesn't contain the `LDAPHierarchicalAuthentication` class anymore. This functionality is now supported by `LDAPAuthentication`, which uses the same configuration options.

If your users are spread out across a hierarchical tree on your LDAP server, you may wish to have DSpace search for the user name in your tree. Here's how it works:

1. DSpace gets the user name from the login form
2. DSpace binds to LDAP as an administrative user with right to search in DNs (LDAP may be configured to allow anonymous users to search)
3. DSpace searches for the user name as within DNs (username is a part of full DN)
4. DSpace binds with the found full DN and password from login form
5. DSpace logs user in if LDAP reports successful authentication; refuses login otherwise

Configuring Hierarchical LDAP Authentication

Hierarchical LDAP Authentication shares all the above standard LDAP configurations, but has some additional settings.

You can optionally specify the search scope. If anonymous access is not enabled on your LDAP server, you will need to specify the full DN and password of a user that is allowed to bind in order to search for the users.

Configuration File:	**`[dspace]/config/modules/authentication-ldap.cfg`**
Property:	`authentication-ldap.search_scope`
Example Value:	`authentication-ldap.search_scope = 2`

Informational Note:	This is the search scope value for the LDAP search during autoregistering (`autoregister=true`). This will depend on your LDAP server setup, and is only really necessary if your users are spread out across a hierarchical tree on your LDAP server. This value must be one of the following integers corresponding to the following values:
	`object scope : 0`
	`one level scope : 1`
	`subtree scope : 2`
	
	Please note that "`search_context`" in the LDAP configurations must also be specified.

Property:	`authentication-ldap.search.anonymous`
Example Value:	`authentication-ldap.search.anonymous = true`
Informational Note:	If true, DSpace will anonymously search LDAP (in the "`search_context`") for the DN of the user trying to login to DSpace. This setting is "false" by default. By default, DSpace will either use "`search.user`" to authenticate for the LDAP search (if search.user is specified), or will use the "`object_context`" value to create the user's DN.
Property:	`authentication-ldap.search.user` `authentication-ldap.search.password`
Example Value:	`authentication-ldap.search.user = cn=admin\,ou=people\,o=myu.edu` `authentication-ldap.search.password = password`
Informational Note:	The full DN and password of a user allowed to connect to the LDAP server and search (in the "`search_context`") for the DN of the user trying to login. By default, if unspecified, DSpace will either search LDAP anonymously for the user's DN (when `search.anonymous=true`), or will use the "`object_context`" value to create the user's DN. **NOTE:** As of DSpace 6, commas (,) are now a special character in the Configuration system. Therefore, be careful to escape any required commas in this configuration by adding a backslash (\) before each comma, e.g. "\,"

IP Authentication

Enabling IP Authentication

To enable IP Authentication, you must ensure the `org.dspace.authenticate.IPAuthentication` class is listed as one of the AuthenticationMethods in the following configuration:

Configuration File:	**`[dspace]/config/modules/authentication.cfg`**
Property:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod`

Example Value:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.IPAuthentication`

Configuring IP Authentication

Configuration File:	**`[dspace]/config/modules/authentication-ip.cfg`**

Once enabled, you are then able to map DSpace groups to IP addresses in `authentication-ip.cfg` by setting `ip.GROUPNAME = iprange[, iprange ...]`, e.g:

```
authentication-ip.MY_UNIVERSITY = 10.1.2.3, \                 # Full IP
13.5, \                      # Partial IP
11.3.4.5/24, \               # with CIDR
12.7.8.9/255.255.128.0, \    # with netmask
2001:18e8::32                # IPv6 too
```

Negative matches can be set by prepending the entry with a '-'. For example if you want to include all of a class B network except for users of a contained class c network, you could use: 111.222,-111.222.333.

Notes:

- If the Groupname contains blanks you must escape the spaces, e.g. "Department\ of\ Statistics"
- If your DSpace installation is hidden behind a web proxy, remember to set the `useProxies` configuration option within the 'Logging' section of `dspace.cfg` to use the IP address of the user rather than the IP address of the proxy server.

X.509 Certificate Authentication

Enabling X.509 Certificate Authentication

The X.509 authentication method uses an X.509 certificate sent by the client to establish his/her identity. It requires the client to have a personal Web certificate installed on their browser (or other client software) which is issued by a Certifying Authority (CA) recognized by the web server.

1. See the HTTPS installation instructions to configure your Web server. If you are using HTTPS with Tomcat, note that the `<Connector>` tag *must* include the attribute `clientAuth="true"` so the server requests a personal Web certificate from the client.
2. Add the `org.dspace.authenticate.X509Authentication` plugin `first` to the list of stackable authentication methods in the value of the configuration key `plugin.sequence.org.dspace.authenticate.AuthenticationMethod`

Configuration File:	**`[dspace]/config/modules/authentication.cfg`**
Property:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod`
Example Value:	`plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.X509Authentication` `plugin.sequence.org.dspace.authenticate.AuthenticationMethod = org.dspace.authenticate.PasswordAuthentication`

Configuring X.509 Certificate Authentication

Configuration File:	**`[dspace]/config/modules/authentication-x509.cfg`**

1. You must also configure DSpace with the same CA certificates as the web server, so it can accept and interpret the clients' certificates. It can share the same keystore file as the web server, or a separate one, or a CA certificate in a file by itself. Configure it by *one*of these methods, either the Java keystore

   ```
   authentication-x509.keystore.path =  path to Java keystore file
   authentication-x509.keystore.password =  password to access the keystore
   ```

   ...or the separate CA certificate file (in PEM or DER format):

```
authentication-x509.ca.cert =  path to certificate file for CA whose client certs to accept.
```

2. Choose whether to enable auto-registration: If you want users who authenticate successfully to be automatically registered as new E-Persons if they are not already, set the `autoregister` configuration property to `true`. This lets you automatically accept all users with valid personal certificates. The default is `false`.

TODO: document the remaining authentication-x509.* properties

Example of a Custom Authentication Method

Also included in the source is an implementation of an authentication method used at MIT, *edu.mit.dspace.MITSpecialGroup*. This does not actually authenticate a user, it *only* adds the current user session to a special (dynamic) group called 'MIT Users' (which must be present in the system!). This allows us to create authorization policies for MIT users without having to manually maintain membership of the MIT users group.

By keeping this code in a separate method, we can customize the authentication process for MIT by simply adding it to the stack in the DSpace configuration. None of the code has to be touched.

You can create your own custom authentication method and add it to the stack. Use the most similar existing method as a model, e.g. `org.dspace.authenticate.PasswordAuthentication` for an "explicit" method (with credentials entered interactively) or `org.dspace.authenticate.X509Authentication` for an implicit method.