# Inc-Atlassian-Options

These comments refer to Confluence VERSION 2.X ?????

## 1) authN -- standard options + various plugins that are available

### a) built-in authN (form within Confluence) -- HTTP authentication with Seraph

http://confluence.atlassian.com/display/CONF256/HTTP+authentication+with+Seraph

Supported authentication methods

The default Seraph authenticator supports four methods of authentication, as can be seen in the flowchart:

* request parameters: os_username and os_password
* session attribute storing the logged-in user
* cookie storing username and password ('remember me' login)
* HTTP basic authentication via standard headers.

Each method is tried in the order above. A successful login at an earlier method continues without checking the later methods. Failure at one method means continuing with the later methods until all are exhausted. At this point, the user is considered an anonymous user, and treated according to the permissions of an anonymous user in Confluence.

Looking through the source code will show that Seraph supports role-based authentication, but this is only used in Confluence for the */admin/* URL restriction.

### b) ldap based User Mgmt - http://confluence.atlassian.com/display/DOC/LDAP+User+Management

Confluence pops up a web form; user enters userid + password; Confluence searches ldap for that user object, then BINDs to ldap as that user. authZ is done via membership in ldap groups. Can co-exist with groups inside Confluence.

### c) REMOTE_User (Shibboleth) -- Shibboleth Authenticator for Confluence http://confluence.atlassian.com/display/CONFEXT/Shibboleth+Authenticator+for+Confluence

Something external to Confluence (web server front-ending the servlet container; plugins in that web server (eg Shibboleth), servlet container configured to do container-based authN) sets the value of the REMOTE_USER variable. This authenticator "trusts" that the external entity has successfully authenticated the browser user, and accepts the user identity presented as the REMOTE_USER value. Can map Shibboleth-delivered attributes to Confluence group names. By default, uses Confluence-resident groups for authZ; see "using ldap groups for authZ, when not using ldap authN".

### d) others (Crowd, CAS, etc)

### e) Issues

1. chaining authenticators.....
2. logout interact with Web SSO (Apache config), rather than builtin mechanism
3. When authentication fails, a recipe is need (and perhaps interfaces?) describing how an authN plugin should return information information to the browser user about the failure and "next steps".

## 2) authZ (authorization)

### a) Confluence-based groups

This is the default mode for the product. Groups are created/managed within Confluence. Users defined within Confluence can be added to Confluence-based groups. Access to spaces + pages can be granted to Confluence-based groups and users.

### b) ldap based User Mgmt - http://confluence.atlassian.com/display/DOC/LDAP+User+Management

After authenticating the browser user via ldap, Confluence will enforce authZ by looking at both Ldap-based groups and Confluence-based groups.

### c) using ldap groups for authZ, when not using ldap authN

(comments from psu) We use the RemoteUserAuthenticator code from Georgetown to authenticate the user using the REMOTE_USER variable.

To enable this and override confluence's built in authenticator, open /var/confluence-2.5.4-std/confluence/WEB-INF/classes/seraph-config.xml and replace the default authenticator of

<authenticator class="com.atlassian.confluence.user.ConfluenceAuthenticator"/>

with

<authenticator class="edu.georgetown.middleware.confluence.RemoteUserAuthenticator"/>

### d) Issues

## 3) User and Group Management

### a) Default user management

Users and groups are stored within Confluence. Web screens/forms are available to create and manage users and groups. A SOAP-Based provisioning interface can be used byan  external provisioning process to manage users and groups.

### b)  ldap based User Mgmt - http://confluence.atlassian.com/display/DOC/LDAP+User+Management

Confluence provides screens allowing an authorized individual to create and manage groups that are defined within Confluence. These groups can contain users defined within Confluence as well as users described within ldap. The standard screens do not allow a browser user to edit the membership of groups defined within ldap. The standard support does not allow a group to have another group as a member.

### c) LDAP Dynamic Groups Plugin

http://confluence.atlassian.com/display/CONFEXT/LDAP+Dynamic+Groups+Plugin (managing groups within Confluence; reflecting changes back to ldap)

This plugin will "synch" group memberships between Confluence and ldap. If a change is made to a Confluence-based group, this plugin will synch the change back to ldap.

### d) Automatically setting the "Can Use" property.

Over the span of several releases, there have been various approaches to the problem of automatically setting this property. There was a plugin:

Automatically Adding LDAP users to the confluence-users Group http://confluence.atlassian.com/display/DOC/Automatically+Adding+LDAP+users+to+the+confluence-users+Group

but this fell out of synch with Confluence APIs at v2.5.6.

With vXXX, there is an administrator setting that allows granting the "Can Use" permission to members of a group. At Brown, we grant this permission to our ldap resident "Community.ALL" group. Our normal provisioning processes manage the membership of this group, and, thus, the ability to use Confluence.

## 4) Managing Access

### a) DEFINE -- setting a localized default ACL on newly created spaces -- this is for SPACES, not PAGES

psu developed this modification:

By default when creating a space, the default ACL only lets the local group "confluence-users" be able to view and modify the new space. Since we use LDAP for authorization and don't want to have a maintain a separate local group, some modification were made to make the default ACL use the LDAP group psu.facstaff.

### b)  globally,  block anonymous users from editing (prevent space owner from turning this back on)

psu developed this mod:

Not allowing anonymous to edit.

Only registered users are allowed to edit pages. Person must at least be logged in with FPS to make changes.
./conf-webapp/src/main/webapp/spaces/includes/createspace_permissions.vm

Comment out the following like so:

```
###           #if ($permissionHelper.globalAnonymousAccessEnabled)
###              #tag( Checkbox "label='create.space.permissions.anonymous'"
###                    "name='permissionSetter.anonymousCanEdit'"
###                    "value=permissionSetter.anonymousCanEdit"
###                    "theme='notable'" )
###           #end
```

Comment out code in ./confluence/conf-webapp/src/main/webapp/template/includes/macros.vm that allows anon to do anything other than view around line 1133.

### c) viewing problem (If you allow anonymous to view a space but don't explicitly give userids/groups permission to view that space, then only anonymous will be able to view the space -- NOT logged in users.)

psu developed a fix for this; However, Atlassian fixed this around v2.5.3.

### d) Permissions Mgmt screen should have a "Select All" button so the administrator can easily give all permissions to a user/group (rather than checking ten individual boxes).

### e) Issues

1. The pages for setting up ACLs and looking up groups break down completely with large numbers of ldap-resident groups.
2. Need documentation describing how to override (or remove) the user/group management pages.
3. Permissions cascade down through a space. It is sometimes necessary, however, to give a person access to a single page at the bottom of the "space tree".

## 5) Miscellaneous

### a) Provisioning

This page describes the APIs that Confluence exports via SOAP and XML-RPC:

http://confluence.atlassian.com/display/DOC/Remote+API+Specification

It appears that all of the required functionality is available.

### b) Change from  Australian to American spellings

psu has developed a patch for this.