Setting up SCIM in PingDirectory 8.x

In PingDirectory 8.x we support SCIMV2 and SCIMV1.1. Since the PingDirectory comes with People, these examples use ou=people,dc=example,dc=com and ou=groups,dc=example,dc=com to create, update, search, and delete Directory Objects.

The following things are included in this docment:

1) Command line to create a Directory
2) The dsconfig commands turn on SCIM. This is included in a dsconfig batch file
3) Ldapmodify commands to create groups and setup the ACI's
4) Debug commands you may need to turn on additional logging
5) Configuration for PingFederate
6) Postman collection to call the API to show off SCIM V1.1 and SCIM V2
7) Notes to assist you
8) Documentation I found useful while testing and debugging

Setup PingDirectory:

**** You need PingDirectory.lic and password-file in the directory you run setup from. password-file will be both your encryption key and the password to the directory**

```
./setup --acceptLicense \
    --licenseKeyFile ./PingDirectory.lic \
    --baseDN dc=example,dc=com --sampleData 100 --localHostName localhost \
    --skipHostnameCheck \
    --allowWeakRootUserPassword \
    --ldapPort 389 --rootUserDN cn=dmanager \
    --rootUserPasswordFile ./password-file --maxHeapSize 384m --primeDB \
    --ldapsPort 636 --httpsPort 8443 --instanceName prod \
    --location Frisco --enableStartTLS --generateSelfSignedCertificate \
    --encryptDataWithPassphraseFromFile ./password-file --no-prompt
```

```
setup --acceptLicense \
    --licenseKeyFile /Users/cmuir/Desktop/final/PingDirectory/PingDirectory.lic \
    --baseDN dc=example,dc=com --sampleData 100 --localHostName localhost \
   --skipHostnameCheck \
    --allowWeakRootUserPassword \
    --ldapPort 389 --rootUserDN cn=dmanager \
```

```
--rootUserPasswordFile password-file --maxHeapSize 384m --primeDB \
--ldapsPort 636 --httpsPort 8443 --instanceName Prod --location Prod \
--enableStartTLS --generateSelfSignedCertificate \
--encryptDataWithPassphraseFromFile password-file --no-prompt
```

- You need to move a license file (PingDirectory.lic) to the root directory of your install, and fix the setup command above to point to your directory.
- You need a file named password-file, with your password and passphrase to encrypt the data. If you are using the command above to create the directory

Files Needed

I have stored the dsconfig batch file here (config-audit.log) download this to the config directory of your PingDirectory 8 install. There are comments in the file of what needs to be changed. Search for "** Change" (this assumes dc=example,dc=com, but you will still have to change the pingfederate location in the file).

**** You must Edit config-audit.log:**

- Search for "sso.pingdemo.org" and replace it with your PingFed server. You can use the one in here for testing, but no guarantee it will be up.
- Search for "dc=example,dc=com", if you need to change the DN of your directory

Download Here the Postman collection that calls the SCIMV2 API to CRUD users and groups.
Download Here the Postman collection that calls the SCIMV1.1 API to CRUD users and groups.
Download Here the Postman Environment variables for Postman

dsconfig command to setup SCIMV2:

- This uses cn=dmanager bind to the directory. Change it in the command below, if you are using a different dn.
- You will need to change the password to match your password.
- You should update tools.properties in the config under your main PingDirectory folder to match your PingDirectory environment. Here is an example of mine.
    - hostname=localhost
    - port=636
    - useSSL=true
    - # useStartTLS=false
    - # trustStorePath=config/truststore
    - bindDN=cn=dmanager

- bindPassword=password

bin/dsconfig --bindDN cn=dmanager --bindPassword password --trustAll --no-prompt --batch-continue-on-error --batch-file config/config-audit.log

If you would like to also set up SCIM V1.1

```
bin/dsconfig set-connection-handler-prop \
   --handler-name "HTTPS Connection Handler"  \
   --add http-servlet-extension:SCIM
```

Create the OU named “groups”

- Example using ldapmodify interactive mode:

```
bin/ldapmodify -a

dn: ou=Groups,dc=example,dc=com
objectclass: top
objectclass: organizationalUnit
ou: groups
```

Set the ACI and scope for these new OU’s created above

```
bin/ldapmodify -a

dn:ou=groups,dc=example,dc=com
changetype:modify
add:aci
aci:(targetattr="*")(version 3.0; acl "ACI for global access"; allow (all)
oauthscope="<admin-scope>";)

dn:ou=people,dc=example,dc=com
changetype:modify
add:aci
aci:(targetattr="*")(version 3.0; acl "ACI for global access"; allow (all)
oauthscope="<admin-scope>";)
```

Example using the scope “admin”

dn:ou=groups,dc=example,dc=com
changetype:modify
add:aci
aci:(targetattr="*")(version 3.0; acl "ACI for global access"; allow (all) oauthscope="admin";)

dn:ou=people,dc=example,dc=com
changetype:modify
add:aci
aci:(targetattr="*")(version 3.0; acl "ACI for global access"; allow (all) oauthscope="admin";)

*** change "<admin-scope>" to a real scope, and set this in environment variable “scope”

Encryption is required, if not already set on PingDirectory:

bin/encryption-settings create --cipher-algorithm AES --key-length-bits 128
--prompt-for-passphrase --set-preferred

Turn on debugging for SCIM:
Note: It is going to ask you how to trust the Cert, choose option 1 ” Automatically Trust”.
Then hit “F” to finish the command.

bin/dsconfig --trustAll --no-prompt set-log-publisher-prop --publisher-name "Debug Trace Logger" --set enabled:true --add scim-message-type:error

Turn on Debugging for ACI

```
bin/dsconfig  --trustAll --no-prompt set-log-publisher-prop \
  --publisher-name "Debug ACI Logger" \
  --set enabled:true
```

Setting up PingFederate

- Create an oauth client named “im_client”, give it a client secret (I used 2Federate)
- Redirect URI:
    - https://www.getpostman.com/oauth2/callback

- Make sure “bypass authorization approval” is checked
- Check implicit under “allowed grant types”

Directions for Postman

- Import the 3 files into Postman (**import** button on the top left)

SCIM V2

- SCIMV2 requires a bearer token, so you will need to make the “Get Bearer Token” call first. We store the bearer token in an Postman environment Variable called {{auth_token}}. Since this is calling PF, and it wants the user to authenticate using the login form.
    - We need to on the “Get Bearer Token” request, you must update the headers to a valid user and password in your directory.Look at the Body of the request we are using pf.username and pf.pass to satisfy this request. The example uses af@example.com and Password1.
    - To prevent it from redirecting
        - Go to Postman menu - preferences (upper left on Mac)- General Tab and uncheck “Automatically follow redirects”

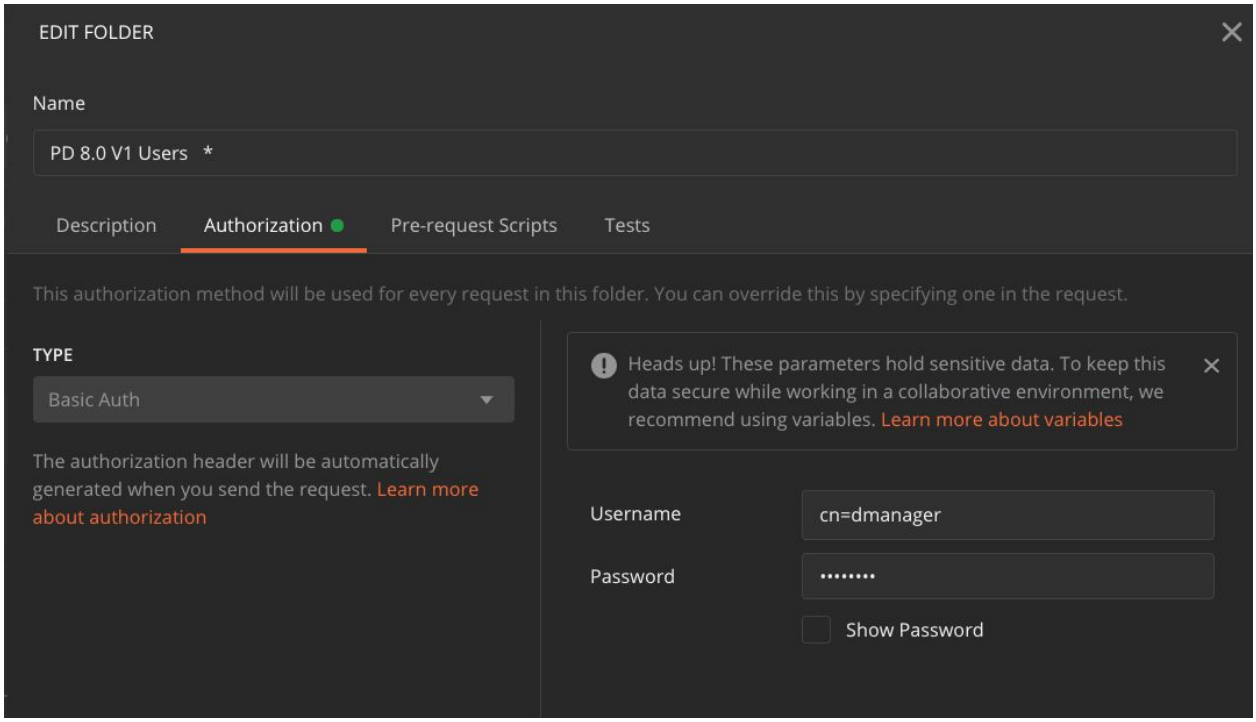
SCIM V1

- SCIM V1 uses basic authentication for the API calls.
    - Update the Authorization on the group for “PD 8.0 SCIMV1 - Users“
        - right-click on the folder “PD 8.0 V1 Users”, choose edit.
        - Choose the authorization tab.
        - Enter your directory admin credentials.
    - Do the same for the group “PD 8.0 SCIMV1 - Groups”

Environment Variables

- The Postman environment variables need to be updated under Manage Environments.

  The collection is called “PingDirectory SCIMV 1.1 and V2” (this was imported earlier)
- Go to the upper right in Postman and click the Gear icon.
- Update the following environment variables, to match your environment:
    - Url
    - pingdirectory_https_port
    - Pingdirectory_port
    - Pingfed
    - pfBaseURL



Example showing updating Basic Auth for SCIM V1

Running the API in Postman

Operations for SCIMV2

For manipulation of resources, SCIM provides a REST API with a rich but simple set of operations, which support everything from patching a specific attribute on a specific user to doing massive bulk updates:

- Create: POST https://example.com/{v}/{resource}
- Read: GET https://example.com/{v}/{resource}/{id}
- Replace: PUT https://example.com/{v}/{resource}/{id}
- Delete: DELETE https://example.com/{v}/{resource}/{id}
- Update: PATCH https://example.com/{v}/{resource}/{id}
- Search: GET https://example.com/{v}/{resource}?filter={attribute}{op}{value}&sortBy={attributeName}&sortOrder={ascending|descending}
- Bulk: POST https://example.com/{v}/Bulk

PD V2 SCIM - Users - Postman Group

Get Bearer Token - This calls PingFederate and gets a bearer token. We are using implicit grant type, in our example. You must have the correct scope, and this must be set on the ACI. This was done above using the scope “admin”

Get All users - Does just that, get all users using the /Users endpoint. This endpoint is defined in “SCIM resource types”

ResoureceTypes - Here you can list the resource types as defined in “SCIM resource types”

ResourceTypes for users - This will show you just the one endpoint.

Get a user - This uses the /Users endpoint and take the id of the user to retrieve it. In our example, that is the field “id”. This is set as the environment variable {{user_id}} in Postman

Find a User with filter ID - This call to the /Users endpoint uses a filter “id eq “{{user_id}}”. You can see more about filters and examples in the 1.1 spec : https://tools.ietf.org/html/draft-scim-api-01#section-3.2.2.1

Find a User with filter mail - Filtering on “mail” using a query parameter.

Create a User - This call uses the JSON in the body of the request to create the user. CN is a required field, and is what is checked for uniqueness. You will get a “409 conflict” return code, if not unique.

Update User - This is a PATCH operation, allowing me to change a specific attribute. This uses the JSON in the body to do the patch. It updates the user represented by the ID in the api call.

Delete User - deletes the user represented in the id ({{user_id}}) in the API call.

PD V2 SCIM - Groups - Postman Group

Create Group - This uses a POST to the /Groups endpoint, This endpoint is defined in "SCIM resource types". The Body of the request has the JSON to create the group. Here cn and uniqueMember are required fields. These will be stored in ou=groups, dc=example,dc=com OU. Since PingDirectory does not come with groups out of the box, we have to create one first, before doing queries. * You can create more than one, cn must be unique.

Get Group - all groups - This will retrieve all groups using the /Groups endpoint

Get Group by id - You will want to use get the id from the "Get Group - all groups" and update your Postman environment variable {{group_id}} before making the call.
"id": "905e9ff8-745a-42f7-9356-64b16793f25a",

Update Group - This call uses the id in the Patch call, and uses the body of the request for which attributes get updated.

Delete Group - This calls a Delete call and deletes the group represented by {{group_id}}

Notes:

- config-audit.log is assumed in the config directory of your PingDir.
- Assume baseDN of dc=example,dc=com for the directory
- This assumes users are stored in ou=people,dc=example,dc=com
- This will create groups in ou=groups,dc=example,dc=com
- SCIMV2 requires that you use a Bearer token. In the config-audit.log , you will need to point to your PingFederate server. There is a Postman call to get your Access Token, but you can also use OauthPlayground.
- On the Patch commands it returns code 204, which is a valid response. Per the SCIM specification, either 204 or 200 are valid return codes.
- This is storing the Bearer Token in a Postman environment Variable named {{auth_token}} so that you do not need to copy and paste.
- You should update tools.properties in the /config folder, here is an example
    - hostname=localhost

- port=636
- useSSL=true
- # useStartTLS=false
- # trustStorePath=config/truststore
- bindDN=cn=Directory Manager
- bindPassword=password

Documentation

- **Ping Identity SCIM 1.1 Deverlopers Guide**
  - **https://www.pingidentity.com/developer/en/resources/scim-1-1-developers-guide.html**
- **IETF SCIM 1 standards document (good examples)**
  - **https://tools.ietf.org/html/draft-scim-api-01**
- **Ping Identity Troubleshooting ACI evaluation**
  - **https://docs.pingidentity.com/bundle/pingdirectory-73/page/kyl1564011473317.html**
- **Information on SCIM V**
  - **http://www.simplecloud.info/**