NCIP

Steps to setting up MOD-NCIP (also see the MOD-NCIP readme file in github)

Step #1 - create a user with the required permissions

- Create User
- Use the reset password link to create a password

The permissions for this user will have to be granted using API calls. These permissions are not in the UX (that I have seen). The last four on this list are actually part of the ncip.all set of permissions... but I think somewhere along the way module permissions might be missing from other modules & granting them to the NCIP user is a work-around. I'm not entirely sure about this... something I need to look into. At this point, the user needs to be granted all of these permissions. I've included a python script I've been using to grant permissions (below) during testing.

ncip.all
inventory-storage.items.collection.get
ui-circulation.settings.overdue-fines-policies
ui-circulation.settings.lost-item-fees-policies
automated-patron-blocks.collection.get

- If you are using the edge-ncip module to allow external access to mod-ncip you will have to create a key that relates to this user. See the edge-common module for more details. What you will end up with is an API key you can use to represent this user. You will be able to provide whomever will be calling your NCIP module externally with a link that will look something like this: http://thedirecturltomyexternalncipmodule/ncip?apikey=eyJzIjoiSkFucGg0Wmw2OFd4ZkRBdnabcedefsgaJ1JiobXsdWKimO The key can also be sent through in the header of the requests. The edge-common module documentation describes the ways you can use the apiKey. The apiKey is the method of authentication for external users of your NCIP module.

Step #2 - Decide on and set up configuration values.

When an NCIP request is sent, it should contain a ‘fromAgency’ value. That value identifies who the request is from. Mod-NCIP in FOLIO requires you to set up configuration values for EACH ‘fromAgency’ you will be accepting requests from. This allows you to set up different configuration values for each ‘agency’ you are accepting NCIP requests from. The values you have to decide on (per agency) are:

- Instance type name: Settings>Inventory>Instances>Resource Types (When the AcceptItem service creates an instance, it will use this type)
- Instance source: You can see this value on this Instance record under Source. It doesn’t have to be created in settings. It appears to be a field you can put any value into. (The AcceptItem service will use this value when it creates the instance)
- Item material type name: Settings>Inventory>Items>Material Types (The AcceptItem service will use this value when it creates the item)
- Item loan type name: Settings>Inventory>Items>Loan Types (The AcceptItem service will use this value when it creates the item)
- Item status name: I use Available (The AcceptItem service will use this value when it creates the item)
- Item permanent location name: Use a location from Settings>Tenant>Locations (The AcceptItem service will use this value when it creates the item)
- Holdings permanent location name: Use a location from Settings>Tenant>Locations (The AcceptItem service will use this value when it creates the holding record)
- Instance custom identifier name: Settings>Instances>Resource Identifier Types

Notes about this: When the ‘AcceptItem’ NCIP service is called, the FOLIO module creates an instance, holding and item record (and page request). The request ID value is set as a Resource Identifier on the instance. This makes searching for the request in the inventory module simple. You can just use the request ID:

- Checkout service point code: Settings>tenants>service points (used when the NCIP Checkout service is called)
- Checkin service point code: Settings>tenant>service points (used when the NCIP Checkin service is called)
The next three configuration options have default values set within the NCIP module so they are optional:

- Include patron’s physical address in the LookupUser response. (set to true or false - it defaults to false)
- Lookup User response string to represent ‘OK’ to borrow - defaults to ACTIVE
- Lookup User response string to represent ‘NOT OK’ to borrow - defaults to BLOCKED

When you have decided on and setup (where needed) these values in FOLIO you will have set them in mod-configuration using API calls.

There is an **example Python script in github** that will create configuration values and a second script to set them in mod-config. In these examples the value ‘ReShare’ is set as the configName for each of the configuration values. This means that when the NCIP module receives requests with the ‘FromAgencyId’ value of ReShare, will apply these configuration values.

Mod-NCIP includes a utility to validate the configuration values are valid. If you send a GET request to: http://myokapiurl/ncipconfigcheck it will attempt to lookup the values you set in mod-configuration to make sure those values exist in FOLIO (for those that need to exist in FOLIO).

There are example NCIP requests in github (note: they use the fromAgency of Relais) and lots more information about the module in the github readme file.

**Other things to keep in mind:**

- Remember to have a circulation rule that will work for the settings you create above.
- The pickup location that will be included in all ‘AcceptItem’ requests MUST be a service point with **pickup location set to TRUE** in FOLIO.

**grant permissions python script:**

```python
import os.path
from git import *
import git, os, shutil
import shutil
import stat
import subprocess
from random import randrange
from pprint import pprint
import json
import sys
import requests
import psycopg2
import time
import urllib.request, json
from collections import OrderedDict

url = "https://folio-snapshot-load-okapi.aws.indexdata.com"
tenant = "diku"

headers = {"x-okapi-tenant": tenant, "Content-type": "application/json"}

#AUTHENTICATE
user = {}
user["username"] = "diku_admin"
user["password"] = "redactedpassword"
user["tenant"] = tenant
the_data = json.dumps(user)
print(the_data)
response = requests.post(url + "/authn/login", the_data, headers=headers)
token = response.headers["x-okapi-token"]
print(response)

headers = {"x-okapi-tenant": tenant, "Content-type": "application/json","x-okapi-token":token}

#PASTE THE UUID OF THE USER HERE:
theUserId = "ff247fa2-6113-4783-84f2-ea54af60424a"
jsonObject = {}
jsonObject['permissionName'] = 'ncip.all'
the_data = json.dumps(jsonObject)
print(the_data)
response = requests.post(url + '/perms/users/' + theUserId + '/permissions?indexField=userId', the_data, headers=headers)
print(response)
print(response.content)

jsonObject = {}
jsonObject['permissionName'] = 'inventory-storage.items.collection.get'
the_data = json.dumps(jsonObject)
print(the_data)
response = requests.post(url + '/perms/users/' + theUserId + '/permissions?indexField=userId', the_data, headers=headers)
print(response)
print(response.content)

jsonObject = {}
jsonObject['permissionName'] = 'ui-circulation.settings.overdue-fines-policies'
print(the_data)
response = requests.post(url + '/perms/users/' + theUserId + '/permissions?indexField=userId', the_data, headers=headers)
print(response)
print(response.content)

jsonObject = {}
jsonObject['permissionName'] = 'ui-circulation.settings.lost-item-fees-policies'
the_data = json.dumps(jsonObject)
print(the_data)
response = requests.post(url + '/perms/users/' + theUserId + '/permissions?indexField=userId', the_data, headers=headers)
print(response)
print(response.content)

jsonObject = {}
jsonObject['permissionName'] = 'automated-patron-blocks.collection.get'
the_data = json.dumps(jsonObject)
print(the_data)
response = requests.post(url + '/perms/users/' + theUserId + '/permissions?indexField=userId', the_data, headers=headers)
print(response)
print(response.content)