MidPoint Advanced Customization MID-102

Student Lab Guide (SAMPLE)

Evolveum, s.r.o.

Revision 4.0-LTS-B, 2020-05-05

This lab guide is not a standalone document and should be used only for the purpose of this training. If there are any questions during the course related to the content of the training or this lab guide itself, do not hesitate to ask the instructor.

If there are any errors, typos or typographic convention mistakes, please report them to the instructor as well. Thank you.

All labs were tested with the midPoint version used during the training.

We assume you have already installed the prerequisites before this training.

Table of Contents

Module 5: Advanced Security Features

LAB 5-1: Password Hashing and Account Activation

Estimated Time: 45 minutes

In this lab, we will update midPoint configuration to store passwords using hashing instead of encryption and the account activation.

In Eclipse, open:

- `objects/resources/localhost-csvfile-1-document-access.xml`
- `objects/resources/localhost-csvfile-3-ldap.xml`
- `objects/resources/openldap-new-corporate-directory.xml`
- search for comment: **MID-102, LAB 5-1** and uncomment the section in all three files
- save, upload and test the resources

This configuration will generate a random initial password for the account to simulate a situation that an account cannot be created password-less. The password will be later changed by the account activation.

The `CSV-2` resource it left intact as we have already removed the password capability from it earlier.

In Eclipse, open and upload:

- `objects/securityPolicies/example-security-policy-hashing.xml`

In your browser with midPoint, set the policy as global:

- go to **System › System**
- in **Global security policy** click the **Edit** icon
- select **`ExAmPLE Security Policy with Password Hashing`**
- click **Save** button

First we will check how the passwords are stored in `User` objects when encryption is in place.

In Eclipse:

- go to **midPoint › Browse server objects**
- select **`User`** in `Object types` section
- enter `picard` to `Names or OIDs (one per line); or an XML query` section

- click **Search** button
- click the `picard` line in the `Result` section
- click **Show** button
- click **Cancel** button

The editor will be opened with the downloaded `picard` user XML object. The object will be saved as a new file in `scratch/gen` directory outside your project object files.

In Eclipse, in `picard` object:

- scroll down to display `<credentials>` section. It should look similar to:

```
<credentials>
  <password>
    <metadata>
      <createTimestamp>2019-09-23T15:07:27.251+02:00</createTimestamp>
      <creatorRef oid="00000000-0000-0000-0000-000000000002"
relation="org:default"
      type="c:UserType"/>
      <createChannel>http://midpoint.evolveum.com/xml/ns/public/gui/
channels-3#user</createChannel>
    </metadata>
    <value>
      <t:encryptedData>
        <t:encryptionMethod>
         <t:algorithm>http://www.w3.org/2001/04/xmlenc#aes256-cbc</t:algorithm>
(1)
        </t:encryptionMethod>
        <t:keyInfo>
         <t:keyName>Y7IfZlAo8A1VMv4sz+X992s/1uk=</t:keyName>
        </t:keyInfo>
        <t:cipherData>
         <t:cipherValue>
/Qh96i7AvKxQo+Gg+s3KAKFMZMb2+TJ5l8Nan5vR8pQ=</t:cipherValue>
        </t:cipherData>
      </t:encryptedData>
    </value>
  </password>
</credentials>
```

(1) You can see that the password is encrypted using `AES256-CBC` algorithm.

In your browser with midPoint:

- go to **Users › All users**

- edit `picard` user
- make sure there are no projections for this user. If there are any, unassign all assignments and then delete all projections before continuing!
- scroll down to **Password** section
- click **Change** button
- set the password to: **`qwerty12345XXXX`**
- click **Save** button

In Eclipse, in `picard` object:

- right click and select **Transfer-related actions › Reload objects from server**
- the editor will reload the `picard` object from midPoint repository
- scroll down to display `<credentials>` section. It should look similar to:

```
<credentials>
  <password>
    <metadata>
      <createTimestamp>2019-09-23T15:07:27.251+02:00</createTimestamp>
      <creatorRef oid="00000000-0000-0000-0000-000000000002"
relation="org:default"
      type="c:UserType"/>
      <createChannel>http://midpoint.evolveum.com/xml/ns/public/gui/
channels-3#user</createChannel>
      <modifyTimestamp>2019-09-25T15:20:35.885+02:00</modifyTimestamp>
      <modifierRef oid="00000000-0000-0000-0000-000000000002"
relation="org:default"
      type="c:UserType"/>
      <modifyChannel>http://midpoint.evolveum.com/xml/ns/public/gui/
channels-3#user</modifyChannel>
    </metadata>
    <value>
      <t:hashedData>
        <t:digestMethod>
          <t:algorithm>http://prism.evolveum.com/xml/ns/public/crypto/
algorithm/pbkd-3#PBKDF2WithHmacSHA512</t:algorithm> (1)
          <t:salt>JpeZiQ==</t:salt>
          <t:workFactor>10000</t:workFactor>
        </t:digestMethod>
        <t:digestValue>
H6IPFRraIbrN8c9aexQq0DrgHdJ8sFocMFNUPGST+NQ=</t:digestValue>
      </t:hashedData>
    </value>
  </password>
</credentials>
```

(1) You can see that the password is hashed using `PBKDF2 with HMAC SHA512` algorithm.

We will setup the notification for account activation now.

In Eclipse, open the file:

- `objects/misc/sysconfig-notifications-lab-5-1.txt`

  Do NOT import or upload the file to midPoint

- copy the XML content from this file to clipboard as indicated by the leading text in the file - just `<handler> .. </handler>` and its contents

In Eclipse:

- open `systemConfigurations/SystemConfiguration.xml`
- right click and select **Transfer-related actions › Reload objects from server**
- the editor will reload the `SystemConfiguration` object from midPoint repository to the local file (and editor)
- paste the content of the clipboard just before `<mail>` element
- save and upload the `SystemConfiguration` object

We will test the account activation notification for a contractor, as all our employees already have their accounts created in all target systems by using the `Internal Employee` role.

In your browser with midPoint:

- login as `administrator`
- go to **Users › All users**
- edit user `badobi`
- change the password to **`qwerty12345XXXX`** to force hashing (otherwise the password is still reversibly encrypted and the account activation will not be required)
- click **Save** button
- edit `badobi` user again (you need to change the password in a **different** operation than assigning the assignments, otherwise midPoint knows the changed password and will not force the account activation)
- go to **Assignments**
- click **New assignment** button and then click the first button in the second row (tooltip: `New Organization type assignment with default relation`)
- select the organization: **`IT Administration Department`**. Without this assignment, the first stage of approvals would automatically reject the request.
- click **Add** button
- click **New assignment** button again and then click the last button in the second row (tooltip: `New assignment`)
- select the roles:
  - **`Basic user`**
- click **Add** button
- click **Save** button

In your browser with midPoint:

- login as `badobi` user
- go to **Request a role**

- in the Role catalog, click **Application bundles** organization
- click **Add to cart** for the `Secret Projects I` role
- click **Go to shopping cart** button
- fill the **Request comment**: `Please approve, I need to work on the X911 project`
- click **Request** button

The request is being approved. You can click the **show case** link to see the request status. The same information is also displayed in your dashboard accessible by clicking on **Home** menu.

Logout as `badobi` user.

We will now approve the request as `administrator` to speed up the process and get to the account activation, but you could of course use proper approvers

In your browser with midPoint:

- go to **Cases › All work items**
- click the ✔ button in the **Assigning role "Secret Projects I" to user "badobi"** request to approve the first stage
- go to **Cases › All work items** again
- click the ✔ button in the **Assigning role "Secret Projects I" to user "badobi"** request to approve the second stage
- go to **Cases › All work items** again
- click the ✔ button in the **Assigning role "Secret Projects I" to user "badobi"** request to approve the third (last) stage

We will check that the role has been assigned.

In your browser with midPoint:

- go to **Users › All users**
- edit user `badobi`
- click **Assignments** tab
- check that the `Secret Projects I` role is assigned after the approvals have completed
- click **Projections**
- `CSV-1` account has been created, `New Corporate Directory` is not created because of weak construction strength

The accounts provided by the role have been now created, but as midPoint doesn't have access to user's password, the accounts can be either password-less or using a generated password (e.g.

as with our `CSV-1` resource). They are also marked as `proposed` in the Shadow’s `lifecycleState` attribute (This is not displayed in GUI nor in the default notification (yet). Actually the account was created and is fully functional, except the password is unknown to the user.). The accounts which do not use passwords can be created immediately in a standard way.

In VirtualBox:

- go to `/opt/training/midpoint-labs/flatfiles` directory
- check the `csv1.csv` file contents for `badobi` account. The account should be created, with a generated password.
- check the notification redirection file: `/opt/training/midpoint-labs/example-mail-notifications.log`
- you should see the following account activation notification sent to the `badobi` user (the link will be different as it contains user’s `oid` attribute):

```
Tue Apr 07 16:34:01 CEST 2020
Message{to='[ben.adobi@example.com]', cc='[]', bcc='[]',
 subject='[IDM] Activate your account(s) ', contentType='null',
  body='Your accounts was successfully created.
   To activate your accounts, please click on the link bellow.

http://192.168.56.20/midpoint/activate/
accounts?user=f351d504-2410-40fc-a21d-6066e954d8bd

User: Ben Adobi (badobi, oid 6999cf3f-c597-4572-826a-daeafc2bdc0d)
Notification created on: Tue Apr 07 16:34:01 CEST 2020

Account to be activated:
 Resource: CSV-1 (Document Access) (oid 10000000-9999-9999-0000-a000ff000002)
 - First name: Ben
 - Employee Number: b8aa6844-21e4-11e8-9e5d-77443d8b7683
 - Login: badobi
 - Groups:
   - Teleportation
   - Time Travel
 - Last name: ADOBI
 - dis: false

Requester: midPoint Administrator (administrator, oid 00000000-0000-0000-0000-
000000000002)
```

- copy the link **from your notification** to the clipboard

Use a different browser and open the link from the clipboard. A page with **Account activation for user `badobi`** title and a password field will display.

You can use the link to activate one or more accounts.

You need to enter the current password to midPoint for user `badobi` (`qwerty12345XXXX`), which will be used for:

1. authentication to midPoint
2. pushing the password to target systems

After you enter the password, the list of activated accounts will be displayed:

- `badobi on resource 10000000-9999-9999-0000-a000ff000002`

In VirtualBox:

- go to `/opt/training/midpoint-labs/flatfiles` directory
- check the `csv1.csv` file contents for `badobi` account. The account password should be set to midPoint user password.
- check the notification redirection file: `/opt/training/midpoint-labs/example-mail-notifications.log`
- you should see the following notification:

```
Tue Apr 07 16:40:44 CEST 2020
Message{to='[idm@example.com]', cc='[]', bcc='[]',
 subject='[IDM] SUCCESS: account MODIFY operation succeeded for badobi',
contentType='null',
  body='Notification about account-related operation

User: Ben Adobi (badobi, oid c419233d-2517-497a-b07e-3b9c7368c603)
Notification created on: Tue Apr 07 16:40:44 CEST 2020

Resource: CSV-1 (Document Access) (oid 10000000-9999-9999-0000-a000ff000002)
Account: badobi

The account has been successfully modified on the resource. Modified attributes
are:
 - Lifecycle state:
   - REPLACE: active
 - credentials/Password/Value:
   - REPLACE: (protected string)

Channel: http://midpoint.evolveum.com/xml/ns/public/gui/channels-3#user

'}, attachmentsCount: 0
```

Of course, changing password from midPoint is still possible as the password will be accessible in clear text form in memory during the operation:

In your browser with midPoint:

- login as `badobi` user
- go to **Credentials**
- enter the old password: `qwerty12345XXXX`
- enter the new password: `qwerty12345BAD`
- click **Save** button

The password is changed. You can check the `/opt/training/midpoint-labs/flatfiles/csv1.csv` file.

It is interesting to know, that the account activation is not required if the password is being generated/created/changed in the same operation as the account provisioning.

In VirtualBox machine, edit the source HR file:

- `/opt/training/midpoint-labs/flatfiles/source.csv`

Use `vim` editor, or `nano` instead of `vim` if you don't like "vi-like" editor or edit the file remotely (e.g. `WinSCP`):

- copy the line of your `X000999` (Arnold Rimmer)
- append the copied line to the end of the file and change the following values in the `source.csv`:
  - **name**: `000980`
  - **firstName**: `Frank`
  - **lastName**: `Rimmer`
  - **empStatus**: `A`
- save the file and quit (in vim: `ESC ESC :wq ENTER`)

Wait a few seconds and click the **Users › All users** menu in midPoint. You should see your new user created in midPoint with the corresponding organizations and `Internal Employee` role assigned.

In VirtualBox:

- go to `/opt/training/midpoint-labs/flatfiles` directory
- check the accounts created for `X000980` user: they should have the password set to the same value, which is the same value as the password in midPoint.

As the LiveSync synchronization has generated the password, while this password was in memory, it was usable for the provisioning process. The accounts have been created without an explicit activation.

You can check the passwords in:

- notification redirection file: `/opt/training/midpoint-labs/example-mail-notifications.log`
- `/opt/training/midpoint-labs/flatfiles/csv1.csv`
- `/opt/training/midpoint-labs/flatfiles/csv3.csv`
- OpenLDAP by binding as `uid=X000980,ou=people,dc=example,dc=com` with the generated password

Note there is no password for `CSV-2` resource, as we have switched off support for passwords for this resource earlier. You can check the self-service GUI.

In a different browser with midPoint:

- login as `X000980`
- go to **Credentials**
- in **Password propagation** line, click ▾ icon to expand the table
- notice the two resource accounts not supporting password changes:
  - ■ `CSV-2`: because the password capability is disabled
  - ■ `ExAmPLE, Inc. HR Source`: because this is a **source** system with no write capabilities
- logout as `X000980`

Finally, we will check that the password was indeed generated and hashed.

In Eclipse:

- go to **midPoint › Browse server objects**
- select `User` in `Object types` section
- enter `X000980` to `Names or OIDs (one per line); or an XML query` section
- click **Search** button
- click the `X000980` line in the `Result` section
- click **Show** button
- click **Cancel** button

The editor will be opened with the downloaded `X000980` user XML object. The object will be saved as a new file in `scratch/gen` directory outside your project object files.

In Eclipse, in `X000980` object:

- scroll down to `<credentials>` and check the password algorithm:

```
<user xmlns="http://midpoint.evolveum.com/xml/ns/public/common/common-3"
  oid="38a5dc59-0d0e-454c-baa0-a8466f9a2f3b"
  version="6">
   <name>X000980</name>
   <!-- the other elements of the User object -->
   <credentials>
      <password>
         <value>
            <t:hashedData>
               <t:digestMethod>
                  <t:algorithm>http://prism.evolveum.com/xml/ns/public/crypto/
algorithm/pbkd-3#PBKDF2WithHmacSHA512</t:algorithm> (1)
                  <t:salt>wNApUg==</t:salt>
                  <t:workFactor>10000</t:workFactor>
               </t:digestMethod>
       <t:digestValue>
tOfEcTloLyVjF1/SiHTKan0w57sjh0uKIakEx1NQJik=</t:digestValue>
            </t:hashedData>
         </value>
      </password>
   </credentials>
</user>
```

(1) You can see that the password is hashed using `PBKDF2 with HMAC SHA512` algorithm.