Post-authentication configuration

Available since 3.8.1

Sometimes there is a need to force the user to set some additional attributes to his profile. Usually all information are based on the records from HR system or set directly into midPoint by hand of administrator. However, not all information could be present in HR or even administrator doesn't know them. In such cases the post-authentication is the best approach how to collect such data. To enable post-authentication in midPoint it is needed to configure the flow policies. This configuration is described bellow.

Configure post-authentication flow

Since midPoint 3.7 there has been added support for self registration and self post-registration. The self post-registration is very similar to the post-authentication flow. The configuration of both are made in security policy. To enable post-authentication you will need to add the following snippet to the system configuration.

Post-authentication configuration

```
<securityPolicy>
   ....
   <flow>
      <postAuthentication>
         <name>Post authentication</name>
         <requiredLifecycleState>proposed</requiredLifecycleState>
         <displayName>Self Registration</displayName>
         <formRef oid="5e8eed5e-7895-4e8b-9beb-ba21ae9a54c0"
                  relation="org:default"
                  type="c:FormType"><!-- Post authentication form --></formRef>
      </postAuthentication>
   </flow>
   ....
</securityPolicy>
```

The example bellow specifies when to enforce the post-authentication for user. The important setting is in *requiredLifecycleState*. It tells which users are forced to perform post-authentication. Using the example above, the users with the lifecycle state set to the *proposed* are automatically redirected to the post authentication form after the successful login. As the post-authentication form custom defined form which is shown to the user can be specified using formRef as in the example above. If the formRef is not specified, the form is the same as on the user details - basic tab.

Configure lifecycle activation

To successfully authenticate to midPoint user has to provide correct credentials and has to be enabled. The enable/disable status of the user is computed based on different properties, e.g attributes validTo, validFrom, administrativeStatus. However the lifecycle state also plays its role in the computation. If the user has lifecycle state e.g. proposed (and the administrative status is not set to enabled) by default it means that this user is disabled and cannot login to the midPoint and cannot perform any actions. But, with this premise it is not possible to enforce post-authentication for such a user because he is not even able to login to midPoint. Therefore there is a need to configure the lifecycle activation. Basically, according to the configuration as shown bellow we can enforce other lifecycle state of the user.

Lifecycle activation configuration

```
<systemConfiguration>
         .....
         <defaultObjectPolicyConfiguration>
      <type>UserType</type>
      <subtype>employee</subtype>
      <lifecycleStateModel>
         <state>
            <name>proposed</name>
                        <!-- no forcedActivationStatus, changing the default to undefined -->
                        <forcedAssignment>
                    <targetType>RoleType</targetType>
                    <filter>
                        <q:equal>
                            <q:matching>polyStringNorm</q:matching>
                            <q:path>name</q:path>
                             <q:value>post Authentication</q:value>
                          </q:equal>
                       </filter>
            </forcedAssignment>
                        <activeAssignments>false</activeAssignments>
         </state>
         <state>
            <name>draft</name>
                        <forcedActivationStatus>archived</forcedActivationStatus>
         </state>
      </lifecycleStateModel>
   </defaultObjectPolicyConfiguration>
        ....
<systemConfiguration>
```

The configuration above shows the configuration for lifecycle state. In the part *lifecycleStateModel* there are two *state* sections, fist for the lifecycle state proposed which doesn't have any *forcedActivationStatus*. It means that if the user has lifecycle state set to proposed, midPoint evaluates it as if the user was enabled (no forcedActivationState means default activation which in midPoint means enabled). In the second section, if the user's lifecycle state is set to draft, the user is considered to be archived and therefore midPoint evaluates it as disabled. Basically, to activate and enforce the post-authentication for the users the value of *requiredLifecycleState must not have any forcedActivationStatus* or the *forcedActivationStatus has to be set to active*. Only then the user can login to the midPoint and then perform the post-authentication.

In the example above, the configuration part for lifecycle state proposed contains more properties. The property activeAssignments can be used to override activation evaluation for user's assignments. If set to true, it means, that even in the proposed state (in which the user is not legally active) all user's assignments are evaluated as active. If set to false, no user's assignments are taken into the account during authentication and authorization phase. If nothing is set, it is evaluated as if is set to true. Of course, user which is going to perform post-authentication needs to have authorizations. At least, if you want the user let the change nickName or telephoneNumber, you need to add him authorization to perform this change. It is recommended to read about authorization to set your role properly.

The forcedAssignments property is set to false. It means, if there are any assignemnts, midPoint evaluates them as disabled and so user cannot perform post-atuhentication. Therefore there is one additional configuration needed and it is forcedAssignment. In this part, it is specified if there are any special roles which has to be enforced by login. It means that in fact, user doesn't have this role assigned, but while the user is in the proposed state, midPoint will pretend that this role is assigned to him. It is recommended to specify the role for post-authentication. You can put there any authorizations needed for post-atuhentication. After the post-authentication is successful, this role no more "belongs" to the user. After successful post-authentication normal assignments are taken into account.

See also

- Authorization
- Authorization Configuration
- GUI Authorizations