User driven federation - SSO

Introduction

This documentation describes the how to install and configure User Driven Federation for Ubisecure SSO.

User Driven Federation enables the end user to link an existing third-party system credential (such as a social network login or bank login) to a target system. The link is created once, after which the third-party identity can be used for ongoing login.

The linking can be done between any two authentication methods. The linking requires that user can be found from the used user repository (for example Ubisecure Directory or CustomerID main user directory)
CustomerID product also supports User Driven Federation and CustomerID specific configuration instructions are present in the CustomerID product manuals.

NOTE: When following the instructions given in this manual, it is advisable to type the commands manually instead of copy/pasting them. This is because sometimes characters may be lost or modified in the copying process.

System overview

To configure user driven federation:

- Decide what is the local authentication method to be used
- Decide what are the third-party authentication methods to be made available for linking
    - Core SSO engine
- Decide which target applications (agents) will enable user driven federation to be used

System software requirements

- Ubisecure SSO 7.1.0

Installation and configuration

TIP: This guide assumes that both methods to be linked have been installed and are in working condition.

LDAP directory structure

Refer sso-udf.ldif
Replace with the SUFFIX value from Ubilogin/config/settings.cmd

Federation Service configuration

- Service name is *"UbiloginFederationTable"* (choose name freely)
- Expects a single input parameter "*subject*" (as-is, no transformation)

```
dn: cn=UbiloginFederationTable,cn=Services,ou=System,
objectClass: top
objectClass: ubiloginService
cn: UbiloginFederationTable
ubiloginClassname: com.ubisecure.ubilogin.federation.spi.ldap.UbiloginFederationTableFactory
ubiloginServiceInputParameter: subject
```

Federation Table

One object of class ubiloginFederationTable must exist.

- Federation table name is *"FederationTable"*
    - In a multi-tenant environment, multiple tables could be configured with different names.

```
dn: cn=FederationTable,cn=UbiloginFederationTable,cn=Services,ou=System,
objectClass: top
objectClass: ubiloginFederationTable
cn: FederationTable
```

User Mapping Service configuration

- Mapping service name is "*federation*"
    - In a multi-tenant environment, multiple mappings could be configured with different names.

```
dn: cn=federation,cn=Server,ou=System,
objectClass: top
objectClass: ubiloginLDAPURLUserMappingTable
cn: federation
```

Federation Mapping entry

- Calls *"UbiloginFederationTable"* service
- Declares value of "*subject*" to expression ${nameID}
- Use subject ${nameID} for all persistent nameID formats
    - For transient NameIDs other common attributes can be used for linking.

```
dn: cn=9871a6a6-cc57-4133-b9e4-2b3c5b9c90f9,cn=federation,cn=Server,ou=System,
objectClass: top
objectClass: ubiloginServiceReference
objectClass: ubiloginServiceUserMappingEntry
cn: 9871a6a6-cc57-4133-b9e4-2b3c5b9c90f9
ubiloginServiceDN: cn=UbiloginFederationTable,cn=Services,ou=System,
ubiloginServiceInputParameter: subject ${nameID}
```

The nameID can be further customized to meet more specialized requirements. The nameID has following attributes that can be refined using EL syntax:

- value
- format
- nameQualifier
- spNameQualifier
- spProvidedID

In a basic use case the combination of these attributes form a globally unique identity, but if it is known that a specific identity can reliably be constructed with the same unique attribute from multiple authentication providers, it is possible to redefine certain attributes that define the authentication provider's namespace, to a common namespace. Effectively, multiple authentication providers' identities are simultaneously mapped with the same identity. Or to simplify: if a user logs in and creates a federation link with authentication method A, she could log out and log in next using authentication method B and a new federation link would be formed behind the scenes based on identical attribute information provided by authentication method B.

For example, the following configuration would overwrite the authentication method specific default nameID attributes with the configured ones and also replace the default value attribute with a CUSTID method attribute provided by the authentication method itself.

```
ubiloginServiceInputParameter: subject ${nameID.format('hetu').nameQualifier('tupas.group').spNameQualifier
('tupas.group').spProvidedID(method.CUSTID).value(method.CUSTID)}
```

It is worth noting that an identity owner of configured authentication method must not be able to freely modify his or her uniquely identifiable attribute in the authentication service – especially not to that of another user, otherwise if a federation link exists the malicious user could gain entry to somebody else's account.

Authentication Method Configuration

Changes are required to the third-party Authentication Method. The third-party Authentication Method must:

- be attached to a user directory
- contain the LDAP attribute ubiloginLDAPURLUserMappingTableDN which points to the User Mapping Service DN.

```
dn: cn=openid.yahoo.1,cn=Server,ou=System,cn=Ubilogin,
ubiloginLDAPURLUserMappingTableDN: cn=federation,cn=Server,ou=System,
```

Application

For every application where user driven federation will be used, at least two authentication methods must be enabled (local and third-party).
An authorization policy must be defined for the agent. Once the authorization policy has been created, add the attribute:

```
ubiloginConfString: FederationManager.TemplateName federation
```

to the application LDAP object. A special template called *federation* contains User Driven Federation specific instructional texts for the end-user. The on-screen instructions will come from the UI messages defined in the named template (in this case *federation*).

For access control, the *Allowed To* tab should contain only groups that contain directory users.

In a multi-tenant environment, different applications could have different UI templates that are configured to match the instructions required for the chosen linkable IDPs.

User interface settings

Language keys for User Driven Federation

Listing 1. UDF language keys in i18n/uas.properties

```
CONFIRM_INTRO_TITLE = Create Account Link
CONFIRM_INTRO_TEXT = Before entering the requested service you can link your external identity with your
existing user permanently.
CONFIRM_HELP_TITLE = Help
CONFIRM_HELP_TEXT = The account you used has not been linked to your existing account. Please save the link and
continue to the service.
CONFIRM_HELP_LINKS =
CONFIRM_LOGIN_TITLE = Account Settings
CONFIRM_LOGIN_TEXT = Please select to remember the account link.
CONFIRM_LOGIN_PERSISTENT_TEXT = Remember this next time
```

Listing 2. UDF language keys in i18n/errors.properties

```
FEDERATION_MISSING = No federated account found
```

Listing 3. UDF language keys in template/messages/federation.properties

```
MENU_LOGIN_TITLE = Already have an account?
MENU_LOGIN_TEXT = Please enter your existing username and password.
MENU_HELP_TITLE = Help
MENU_HELP_TEXT = The account you used to log in hasn't been used before at this service. If you already have an
existing local account, please sign in with your username and password above to link your account. Please sign
in with your existing username and password or register a new account.
MENU_HELP_LINKS = <li><a href="javascript:view.navigate('register.account')">Register</a></li><li><a href="
javascript:view.navigate('password.reset')">Password Reset</a></li>
PASSWORD_HELP_TITLE = Help
PASSWORD_HELP_TEXT = The account you used to log in hasn't been used before at this service. If you already
have an existing local account, please sign in with your username and password above to link your account. If
you don't have an account, please create one by registering
```

Listing 4. UDF configuration keys in template/default/federation.properties

```
title = Ubisecure SSO
usemethodgroups = true
links = federation.links
```

Listing 5. UDF configuration keys in template/default/federation.links

```
register.account.url = https://cid.example.com:7443/eidm2/wf/register/udf
register.account.methods = password.1 password.2
password.reset.url = https://cid.example.com:7443/eidm2/wf/lostpwd
password.reset.methods = password.1 password.2
```

The configuration above defines to which registration users should be redirected if it is desired for users to be able to create their user accounts during the login process.

Verifying The Installation

To verify that Ubisecure SSO is installed or upgraded successfully:

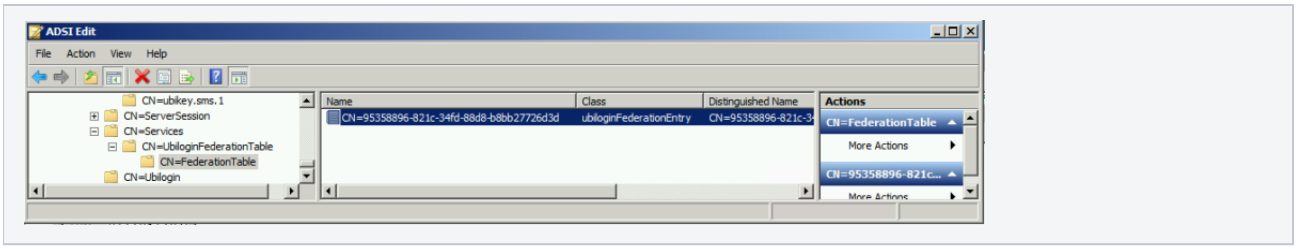
1. Attempt to access the target application.
2. Two options should be available, local and remote.
3. Local login should work and you will be granted access to the application.
4. Logout and attempt to access the target agent again.
5. Choose the third-party login method and login at third-party IDP.
6. Upon return you will see an error message stating that you don't yet have a federation link for the third-party authentication method.
7. Login once using the local account and mark that the link will be saved.
8. Logout and attempt to access the target agent again using the third-party IDP.
9. You will be granted access to the application without further prompting.

Terminating the linking

One UbiloginFederationEntry is created for each account linked.

To break the link between accounts, the matching entries must be deleted from LDAP under `CN=FederationTable,OU=UbiloginFederationTable, CN=Services`.

In future, terminating the linking will be possible by the end user or REST interface.



Preventing disabled users from logging in with user driven federation

Starting from Ubisecure SSO 8.3, user account status for registered users in Ubilogin Directory, Active Directory and SQL Directory is validated during logging in with user driven federation. That means that the authorization policy based workaround instructed for previous versions is not needed anymore.