Upgrading a 2.x IdP to 3.x

Shibboleth IdP 2.x will become End-Of-Life (EOL) on July 31st, 2016. All production IdPs have to be upgraded to version 3.x.

Rationale

While upstream provides documentation on upgrading a 2.x instance in-place to 3.x, our recommendation (especially for Production deployments) is to start a clean install and copy only the attribute resolver configuration. The benefits of a installing on a new system are:

- a clean Shibboleth IdP install (without legacy 2.x configuration entries)
- a clean OS install (we recommend switching to RHEL/CentOS 7)
- ability to test the new v3 install while the old v2 IdP continues operating

The detailed plan below still provides for a smooth user experience (i.e., outage-less upgrade) that would retain user's identities across the upgrade.

Prerequisites

Earlier versions of this manual were instructing when performing an upgrade from a 2.x IdP that was using ComputedIdConnector for eduPersonTargetedID (i.e., not storing the values in a database), to first follow the instructions at Configuring an 2.x IdP to use StoredID Connector.

While it is still strongly recommended to store the values in a database, it is no longer deemed necessary to change the configuration of the version 2 IdP, as the version 3 IdP would be producing the same values.

However, if the existing persistent ID values are not stored in a database, it is crucial to use the identical salt value (as well as other parameters to the hash functions) on the old 2.x and the new 3.x IdP.

Note that as IdP 3.3.x introduces a new setting `idp.persistentId.encoding` and makes this default to BASE32 (while the previous default behavior was to use BASE64 encoding), to get the same persistentID values, it is crucial to explicitly override this setting in `/opt/shibboleth-idp/conf/saml-nameid.properties`:

```
idp.persistentId.encoding = BASE64
```

Upgrade steps

- Provision a new host (VM) with RHEL/CentOS 7, meeting the prerequisites as per the IdP 3.x Installation manual.
  - Pick a new hostname identifying the machine as per local conventions - the user facing hostname will be redirected here later.
- Start installing the IdP, following the IdP 3.x Installation manual - following all steps except the attribute definitions.
  - When configuring the IdP (and running the installer), pick the same external facing hostname and the same entityId as your existing IdP.
  - The same applies to hostname used in web server SSL certificates.
- Configure attribute definitions. We recommend to:
  - Copy over the `attribute-resolver.xml` file from the 2.x IdP.
  - Note that reusing the `attribute-resolver.xml` file also carries over the "Friendly attribute names" - if they have been configured in the 2.x IdP.
  - Make the following changes to the `attribute-resolver.xml` file to reflect the changes in the IdP 3.x setup:
    - Edit the LDAP connector definition and switch from hard-coded connection details to referring to properties in `ldap.properties` (as is done in the sample file `attribute-resolver-ldap.xml` included in the 3.x install.
      This would make the configuration resilient to future changes to LDAP connection details - changing them in `ldap.properties` would be enough.
    - Remove the now deprecated PrincipalConnector elements (mapping NameIDs back to usernames is done differently in IdPv3 and is now covered by the Persistent NameID configuration).
    - Remove (or comment-out) definitions of older NameID attribute and connector definitions:
      - eduPersonTargetedID
      - ComputedId
      - StoredId
      - TransientId
    - Update the `sharedTokenDataConnector` definition:
      - Add a dependency for the source attribute (otherwise would not be available)
      - If storing values in database, add settings for `preferredTestQuery` and `testConnectionOnCheckout`
      - If storing values in LDAP, add `ldapConnectorId` to refer to the LDAP Connector (and still make sure it is included in the dependencies)
- Copy over databases from 2.x IdP:
  - Part of the 3.x setup is creating an `idp_db` database with three tables:
    - The `tb_st` table holds values for the auEduPersonSharedToken attribute
    - The `shibpid` table holds values for eduPersonTargetedID / SAML Persistent NameID
    - The `StorageRecords` table holds various internal IdP data (including user consent, sessions, artifacts, replay cache)
  - The `tb_st` and `shibpid` tables should be copied over from the 2.x to the 3.x IdP as part of the cutover. This can be achieved with:

```
# on 2.x IdP
mysqldump idp_db tb_st shibpid > idp-data.sql

# on 3.x IdP
mysql idp_db < idp-data.sql
```

- Preparation for cut-over:
  - To operate the new 3.x IdP in parallel with the 2.x IdP, register the **signing** key of the new 3.x IdP into the Federation Registry as a additional signing key for your existing IdP.
    - This would make all SPs in Tuakiri accept assertions from both the old 2.x and the new 3.x IdP (assuming they use the same entityId).
  - You can test the new 3.x IdP by going to a URL of the form **https://idp3.inst.ac.nz/idp/profile/SAML2/Unsolicited/SSO?providerId=https://attributes.tuakiri.ac.nz/shibboleth**
    - In this case, you are accessing the IdP via the internal hostname, not the public hostname end-users would see. (So at this stage, it is acceptable to get a certificate mismatch - this is not what the end-users would experience).
  - You can also test the new IdP by pointing your client machine / browser to the new IdP - e.g., by inserting an entry into your `/etc/hosts` file linking the IP address of the new 3.x IdP with the public hostname it is registered with.
  - Service Provider specific parts of the cutover:
    - Thomson Reuters: Thomson Reuters uses different user mappings for eduPersonTargetedID and Persistent NameID. We need to advice them in advance (ideally with a few days lead time) and ask them to change the mapping for the IdP being upgrade. Please either contact them directly at science.ShibbolethSupport@thomsonreuters.com or contact Tuakiri support at tuakiri@reannz.co.nz.

- Cut-over to the new IdP: Once testing is confirmed as per above:
  - Do a final copy of the `tb_st` and `shibpid` tables from the old 2.x IdP to the new 3.x IdP
  - Change the public DNS name of your IdP to point to the new 3.x IdP
  - Or, if you are using a load balancer, just reconfigure your load balancer to use the new 3.x IdP and drop the old 2.x IdP

- Post cut-over: once you have confirmed the upgrade has been successful:
  - Add the encryption key and the back-channel signing key of the new 3.x IdP to your IdP registration in the Tuakiri Federation Registry
    - Adding the encryption key earlier could make some SPs send messages to the 2.x IdP encrypted with a private key only known to the 3.x IdP
  - Declare support for PersistentID NameFormat in the IdP registration in the FR (SAML -> NameIDFormats -> Add `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent)`
  - Remove the signing key of the old 2.x IdP
  - Decommission the old 2.x IdP VM