IDP 4 Attribute release

By default the Shibboleth Identity Provider (IDP) software will not release any data to any service (except for a short-lived, opaque identifier called transient NameID, also there are a few example rules included in the distributed attribute-filter.xml for illustration purposes).

While not sending any data to anyone is "secure" – much the same way as not connecting a computer to a network is "secure" – it's also not very practical, as essentially all Service Providers (SP) need attributes in order to provide their services and/or perform access control.

There are several approaches to enabling controlled release of attributes to the appropriate services and most (or all) of them will probably need to be deployed within an institutional IDP:

As mentioned in the main Service Categories article, be sure to get use of these (or similar) rules within your IT systems approved by your institution's management.

The following policies and rules are meant to be added to your Shibboleth IDP's attribute filter configuration, by default in /opt/shibboleth-idp/conf/attribute-filter.xml

Service Categories

Managing what attributes should be released and to what services can be a daunting task, given the plethora of services available in the eduID at federation, and even more so for Interfederation. Service Categories (also called Entity Categories) have been created to address this very problem by allowing IDPs to perform controlled but still automated attribute release for services that have been assigned (to) one of the standardized categories. This page contains example implementations of the most important Service Categories as attribute filter rules for the current Shibboleth IDP software.

REFEDS Research & Scholarship

The REFEDS R&S service category definition includes an attribute bundle, i.e. a set of attributes that minimally and maximally constitutes what services "that are operated for the purpose of supporting research and scholarship interaction, collaboration or management, at least in part" commonly need and therefore should recieve:

```
<AttributeFilterPolicy id="REFEDSResearchAndScholarship">
  <PolicyRequirementRule xsi:type="EntityAttributeExactMatch"
    attributeName="http://macedir.org/entity-category"
    attributeValue="http://refeds.org/category/research-and-scholarship"/>
  <!-- RandS requires: An identifier, email and a person's name. If ePPN values could be reassigned you MUST also release eduPersonTargetedID/persistent NameID. Always releasing ePTID/persistent NameID is recommended, though. As is releasing givenName+sn in addition to displayName, to help with interoperability. -->
  <AttributeRule attributeID="eduPersonPrincipalName" permitAny="true" />
  <AttributeRule attributeID="eduPersonTargetedID" permitAny="true" />
  <AttributeRule attributeID="mail" permitAny="true" />
  <AttributeRule attributeID="displayName" permitAny="true" />
  <AttributeRule attributeID="givenName" permitAny="true" />
  <AttributeRule attributeID="sn" permitAny="true" />
  <!-- Affiliation is optional but release is still "strongly recommended". -->
  <AttributeRule attributeID="eduPersonScopedAffiliation" permitAny="true" />
</AttributeFilterPolicy>
```

Note if your deployment of the eduPersonPrincipalName (ePPN) attribute allows for reassigning ePPN values from one person to another that R&S v1.x requires you to also release a SAML 2.0 persistent NameID then. Because SAML 2.0 persistent NameIDs prohibit reassignment also releasing those enables R&S Service Providers (that care and do the work) to detect ePPN reassignments by always looking at the combination of ePPN and persistent NameID (instead of just at the ePPN): If one day a known ePPN arrives together with a different/unknown persistent NameID that's a sign that the ePPN likely has been reassigned to a different person and that access to data held at the SP from/about the former ePPN-NameID combination should be denied.

Of course that assumes that you can and do support proper SAML 2.0 persistent NameIDs and that you create those from source data that itself is not reassigned from one person to another. (If you don't have any such data and you also cannot mint it on the fly, as illustrated in our IDP attribute resolver documentation, you cannot produce spec-conformant persistent NameIDs and therefore also cannot support REFEDS R&S v1.)
Also note that if you don’t have the (old, deprecated) `eduPersonTargetedID` (ePTID) attribute configured in your IDP then the Shibboleth IDP attribute filter rules given above may not be sufficient to conform to R&S v1: Technically (proper) SAML 2.0 persistent NameIDs are not "attributes" and consequently cannot be released using the Shibboleth IDP’s attribute filter configuration (contrary to ePTID attributes, which wrap a persistent NameID in an attribute, but which are deprecated and their use should be phased out). Unless a REFEDS R&S Service Provider also signals that it wants to receive SAML 2.0 persistent NameIDs in its SAML Metadata using the `NameIDFormat` element (which other federations don’t even support and many SPs simply don’t currently do) the above attribute filter snippet is insufficient to satisfy the R&S 1.x requirements – if your ePPN values may be reassigned and you don’t have ePTID available.

So (only) if all of the following 3 criteria apply to your deployment:

- ePPN attribute values may be reassigned from one person to another, over time,
- You have some other (internal) data available (or can mint it on the fly, as illustrated in our IDP attribute resolver documentation) that can serve as a basis for conformant (i.e., not-reassigned) attributes and persistent NameIDs,
- You don’t have the (old, deprecated) `eduPersonTargetedID` (ePTID) attribute configured in your IDP,

then you’d need to amend your IDP’s `relying-party.xml` configuration with a “relying party override” like the one shown below, forcing the NameIDFormat to "persistent" for all REFEDS R&S Service Providers (just in case those SPs don’t already request that NameID format in their metadata):

```xml
<util:list id="shibboleth.RelyingPartyOverrides">
  <bean parent="RelyingPartyByTag">
    <constructor-arg name="candidates">
      <list>
        <bean parent="TagCandidate" c:name="http://macedir.org/entity-category"
             p:values="http://refeds.org/category/research-and-scholarship" />
      </list>
    </constructor-arg>
    <property name="profileConfigurations">
      <list>
        <bean parent="SAML2.SSO" p:nameIDFormatPrecedence="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent" />
        <ref bean="SAML2.ECP" />
        <ref bean="SAML2.Logout" />
      </list>
    </property>
  </bean>
</util:list>
```

N.B.: If your `shibboleth.DefaultRelyingParty` setting includes other properties on the bean with parent="SAML2.SSO" you should also include /repeat them here, e.g. `p:postAuthenticationFlows="attribute-release`.

Of course that's overly complex (or plain crazy) which is exactly why ePPN, ePTID, persistent NameIDs and even R&S v1 are all more or less in the process of being replaced or phased out for newer, simpler replacements. For existing services relying on these standards, attributes and NameIDs, though, the above may be needed for proper support by the IDP.

**GÉANT EU/EEA Data Protection Code of Conduct for Service Providers**

The **GÉANT Data Protection Code of Conduct** category definition does not specify an attribute bundle (i.e., it doesn’t reference one set of attributes which should be released to all category members). As such the set of attributes to release may vary from service to service, though the data is still limited to attributes "that are necessary for enabling access to the service provided by the Service Provider" (2.b, "purpose limitation"). The configuration below is an example based on the most commonly used attributes in academic Identity Federations today which all `eduID.at Identity Providers` should be able to generate.
ACOnet-registered services

If you’re satisfied with the checks performed by the ACOnet team during registration of new ACOnet Identity Federation member Service Providers – cf. our Metadata Registration Practice Statement, section 5.4 – you may decide to release any attributes that are listed as RequestedAttribute elements in the Service Provider’s SAML Metadata to all services that have been registered by ACOnet bona fide ACOnet Identity Federation members (which are bound by the ACOnet Identity Federation Policy.) Whether those facts are considered sufficient to release personal data to a service remains a local decision.

![Image](https://via.placeholder.com/150)

While the ACOnet team always tries to negotiate sensible and limited/minimal use of personal data with all Service Providers it registers, the ultimate decision what attributes a service needs (or claims to need) remains with the legal entity representing the Service Provider. Similarly, the decision to actually release attributes remains with the institution sending the data (or the subjects herself, when valid consent was obtained).

As with the “GÉANT Data Protection Code of Conduct” example above this rule establishes an upper limit on the set of attributes you’d be willing to release under the specified conditions. i.e., provided the Service Provider requests the attributes in SAML Metadata, marks them as required (as appropriate), and the service has been registered by the ACOnet team.

You can adjust the maximum set of attributes to be released under this policy rule (by adding or removing AttributeRule elements), as well as whether you’ll only release them if marked “required” or also when they’re merely “requested” (signalling either an optional attribute or an acceptable alternative, the latter being very common and cannot be expressed in SAML 2.0 Metadata properly).
<AttributeFilterPolicy id="RegisteredByACOnetRequiredAttributes">
  <PolicyRequirementRule xsi:type="RegistrationAuthority" registrars="http://eduid.at"/>
  <AttributeRule attributeID="displayName">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"/>
  </AttributeRule>
  <AttributeRule attributeID="givenName">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"/>
  </AttributeRule>
  <AttributeRule attributeID="sn">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"/>
  </AttributeRule>
  <AttributeRule attributeID="mail">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"/>
  </AttributeRule>
  <AttributeRule attributeID="eduPersonScopedAffiliation">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"/>
  </AttributeRule>
  <AttributeRule attributeID="eduPersonPrincipalName">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"/>
  </AttributeRule>
  <AttributeRule attributeID="eduPersonUniqueId">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="true"/>
  </AttributeRule>
  <AttributeRule attributeID="eduPersonTargetedID">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="false"/>
  </AttributeRule>
  <AttributeRule attributeID="schacHomeOrganization">
    <PermitValueRule xsi:type="AttributeInMetadata" onlyIfRequired="false"/>
  </AttributeRule>
  <AttributeRule attributeID="eduPersonEntitlement">
    <PermitValueRule xsi:type="AND">
      <Rule xsi:type="AttributeInMetadata" onlyIfRequired="false"/>
      <!-- Limit entitlement values that can be released under this generic policy. -->
      <Rule xsi:type="OR">
        <Rule xsi:type="Value" value="urn:mace:dir:entitlement:common-lib-terms"/>
        <Rule xsi:type="Value" value="urn:mace:terena.org:tcs:personal-user"/>
        <Rule xsi:type="Value" value="urn:mace:terena.org:tcs:escience-user"/>
        <Rule xsi:type="Value" value="https://rdb.manz.at/student/remote-access"/>
        <Rule xsi:type="Value" value="https://rdb.manz.at/fellow/remote-access"/>
        <Rule xsi:type="Value" value="http://usi.at/student-discount"/>
      </Rule>
    </Rule>
  </AttributeRule>
  <AttributeRule attributeID="eduPersonEntitlement">
    <PermitValueRule xsi:type="AND">
      <!-- Limit entitlement values that can be released under this generic policy. -->
      <Rule xsi:type="OR">
        <Rule xsi:type="Value" value="urn:mace:dir:entitlement:common-lib-terms"/>
        <Rule xsi:type="Value" value="urn:mace:terena.org:tcs:personal-user"/>
        <Rule xsi:type="Value" value="urn:mace:terena.org:tcs:escience-user"/>
        <Rule xsi:type="Value" value="http://usi.at/student-discount"/>
      </Rule>
    </Rule>
  </AttributeRule>
</AttributeFilterPolicy>

SAML Subject-ID Attributes

Recent versions of the Shibboleth IDP software ship with a default/example attribute filter rule that releases the SAML Standard Identifier attributes (SAML Subject-ID, SAML Pairwise-ID) to any known SP that requests them according to the new signalling (using an extension Entity Attribute instead of the traditional RequestedAttribute metadata element). While that's helpful in getting up and running quickly you may want to further limit to whom you're passing along this personal data.

In the examples below we've added alternative requirements just when these identifiers should be released. In the first example shown below either of the requested SAML Subject-ID attributes would only go out to SPs belonging to the REFEDS Research & Scholarship service category (see above or Service Categories) or belonging to the GÉANT Data Protection Code of Conduct service category (ibid.) or having been registered locally by ACOnet (and therefore are bound by the ACOnet Identity Federation policy).

Arguably the safeguards added below are somewhat arbitrary but some may still find this preferrable over releasing personal data to just any SP that's capable of asking for them in the technically correct way.
An alternative implementation of such additional safeguards could apply those only for the "samlSubjectID" attribute but not for "samlPairwiseID": "samlPairwiseID" is much more privacy preserving – at least when not released together with more identifying data – and could therefore possibly be released without such additional safeguards:
### SAML Subject-ID with additional safeguards, alternative 2

```
<AttributeFilterPolicy id="subject-identifiers">
  <PolicyRequirementRule xsi:type="ANY"/>

  <AttributeRule attributeID="samlPairwiseID">
    <PermitValueRule xsi:type="OR">
      <Rule xsi:type="EntityAttributeExactMatch"
          attributeName="urn:oasis:names:tc:SAML:profiles:subject-id:req"
          attributeNameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
          attributeValue="pairwise-id"/>
      <Rule xsi:type="EntityAttributeExactMatch"
          attributeName="urn:oasis:names:tc:SAML:profiles:subject-id:req"
          attributeNameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
          attributeValue="any"/>
    </PermitValueRule>
  </AttributeRule>
</AttributeFilterPolicy>

<AttributeFilterPolicy id="samlSubjectID">
  <PolicyRequirementRule xsi:type="OR">
    <Rule xsi:type="EntityAttributeExactMatch"
      attributeName="http://macedir.org/entity-category"
      attributeValue="http://refeds.org/category/research-and-scholarship"/>
    <Rule xsi:type="EntityAttributeExactMatch"
      attributeName="http://macedir.org/entity-category"
      attributeValue="http://www.geant.net/uri/dataprotection-code-of-conduct/v1"/>
    <Rule xsi:type="RegistrationAuthority" registrars="http://eduid.at"/>
  </PolicyRequirementRule>

  <AttributeRule attributeID="samlSubjectID">
    <PermitValueRule xsi:type="EntityAttributeExactMatch"
      attributeName="urn:oasis:names:tc:SAML:profiles:subject-id:req"
      attributeNameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
      attributeValue="subject-id"/>
  </AttributeRule>
</AttributeFilterPolicy>
```

It is expected we’ll be seeing specific Service Categories being developed to help make better and more scalable use of these new identifiers.

### Enumerating services

For services not (yet) covered by any other methods (see above), enumerating entityIDs in the configuration allows to apply a set of common rules to a list of services sharing certain properties. Here’s an example configuration releasing only basic attributes to selected services that will be useful to most institutions. The Service Providers’ entityIDs (i.e., their globally unique names) are OR’ed together in the PolicyRequirementRule, meaning the policy will apply to any of the services listed.

### Examples: Collaboration and NREN services

```
<AttributeFilterPolicy id="CollaborationAndNRENServices">
  <PolicyRequirementRule xsi:type="OR">
    <Rule xsi:type="Requester" value="https://www.aco.net/shibboleth"/>
    <Rule xsi:type="Requester" value="https://eduroam.aco.net/shibboleth"/>
    <Rule xsi:type="Requester" value="https://wiki.univie.ac.at/shibboleth"/>
    <Rule xsi:type="Requester" value="https://www-vhosts.univie.ac.at/shibboleth"/>
    <Rule xsi:type="Requester" value="https://www03.univie.ac.at/shibboleth"/>
  </PolicyRequirementRule>

  <AttributeRule attributeID="eduPersonPrincipalName" permitAny="true"/>
  <AttributeRule attributeID="displayName" permitAny="true"/>
  <AttributeRule attributeID="mail" permitAny="true"/>
  <AttributeRule attributeID="eduPersonScopedAffiliation" permitAny="true"/>
</AttributeFilterPolicy>
```
Individual policies

Services requiring special configuration are often best dealt with by giving them their own, individual filter policy. Examples can be found in this wiki, e.g.:

- TCS Personal Certs
- USI Wien
- RDB
- ubook

One policy you’ll probably want to add is releasing all locally defined attributes to the eduID.at Demo Service Provider in order to be able to easily check the configuration and attribute values. (Note that the eduID.at Demo SP does not record (or persist) any received attribute values, these are only processed in volatile memory as part of your session.)

An policy releasing all the attributes defined in this documentation to the eduID.at Demo SP could be as simple as this:

```xml
<AttributeFilterPolicy id="eduID.at-Demo-SP">
  <PolicyRequirementRule xsi:type="Requester" value="https://test-sp.aco.net/shibboleth" />
  <AttributeRule attributeID="givenName" permitAny="true" />
  <AttributeRule attributeID="sn" permitAny="true" />
  <AttributeRule attributeID="displayName" permitAny="true" />
  <AttributeRule attributeID="mail" permitAny="true" />
  <AttributeRule attributeID="samlSubjectID" permitAny="true" />
  <AttributeRule attributeID="samlPairwiseID" permitAny="true" />
  <AttributeRule attributeID="eduPersonPrincipalName" permitAny="true" />
  <AttributeRule attributeID="eduPersonScopedAffiliation" permitAny="true" />
  <AttributeRule attributeID="eduPersonEntitlement" permitAny="true" />
  <AttributeRule attributeID="eduPersonTargetedID" permitAny="true" />
  <AttributeRule attributeID="schacHomeOrganization" permitAny="true" />
  <AttributeRule attributeID="schacPersonalUniqueCode" permitAny="true" />
</AttributeFilterPolicy>
```

If you have more attributes defined in your IDP that you also want to be able to see in the eduID.at Demo SP simply add them to the above list of AttributeRule elements. (Note that if those attributes are not widely used the eduID.at Demo SP may not be configured to look for them yet, and therefore may not show them in the tabular overview of received attributes you get as default view after logging in there. You can still see all attributes sent in the view of the SAML assertion, though.)

Local services

An IDP might also have access to Service Providers not registered with the eduID.at federation (or any other, for that matter), which are only available to the institutional IDP. Those Service Providers might be managed in a local file at the IDP containing their SAML metadata. In such a case it is possible (and sometimes practical) to release a certain set of attributes to all Service Providers in that specific local metadata document. This can be achieved by giving the SAML metadata document a name – by setting the Name XML attribute on a surrounding EntitiesDescriptor XML element – and referencing that name by value in the attribute filter policy.

An example metadata file, e.g. in /opt/shibboleth-idp/metadata/local-sps.xml could look like this and would contain locally managed Service Providers as child EntityDescriptor elements:

```xml
<EntitiesDescriptor xmlns="urn:oasis:names:tc:SAML:2.0:metadata" xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" Name="https://example.edu/local-entities">
  <!-- individual EntityDescriptor elements go here, one for each SP -->
</EntitiesDescriptor>
```

An attribute filter policy active for all Services Providers present in that file could then look like this, referencing the Name attribute (value) of the EntitiesDescriptor element (see above) in the groupID attribute of the PolicyRequirementRule (see below). In this (somewhat fictitious) example all Service Providers whose metadata is available in a metadata file with that Name attribute would receive the (only locally relevant or unique) attributes uid, employeeNumber, provided those were defined in the IDP and had any values for the given subject:
Of course the file containing the metadata would need to be registered with the IDP once (in a MetadataProvider, in /opt/shibboleth-idp/conf/metadata-providers.xml) for the IDP to know about those entities. But only that one file with all local SPs, instead of having to add every single SAML SP to the IDP that way (which does not scale and also introduces the problem of having to reload the metadata providers configuration for each SP addition/change/removal). That one metadata provider could be set to reload automatically so new (or changed or removed) SPs contained in that file will be discovered automatically. Though since only you or your own local processes will be updating the content of that local metadata file you could as well reload only that metadata (from the command line or via HTTP/S) specifically after having updated it.

Default/Fallback attribute release

An institution may also decide (in addition to all the above rules) to release a very small set of attributes – possibly even more limited than that of the R&S attribute bundle – unconditionally and to every SP it knows via trusted metadata. That allows the institution's members the use of more services than are currently covered by any of the methods outlined above, with essentially no additional risk due to the innocuous nature of the data involved. A simple rule to implement such an approach could look like this, releasing only eduPersonScopedAffiliation (a person's role or affiliation within an institution, in very coarse standardized terms such as "student" or "faculty") and schacHomeOrganization which only contains the institution's canonical DNS domain.

```
<AttributeFilterPolicy id="InnocuousDataToAnyServiceViaTrustedMetadata">
  <PolicyRequirementRule xsi:type="ANY"/>
  <AttributeRule attributeID="eduPersonScopedAffiliation" permitAny="true"/>
  <AttributeRule attributeID="schacHomeOrganization" permitAny="true"/>
</AttributeFilterPolicy>
```

While it's possible to add more or other AttributeRule elements to such an "open" release policy (thereby also releasing more/other attributes) this should only be done after careful consideration of the consequences. If in doubt contact the ACOnet team first.

Consent

Finally, in other cases where the institution (or rather its management) does not want to be responsible for the release of attributes to services, i.e., in cases where none of the above applies (no Service Category or enumeration covering the SP, but the service needs more attributes than one is willing to release to "anyone", as per above, in order to allow access), the IDP could release any attributes an SP requests (in its SAML Metadata, provided the IDP has the attributes configured locally) but makes the release conditional to the subject's own freely given, informed, revocable consent. This is intended to still allow subjects from an institution get access to a wide range of services they might need, even when the institution has no policy in place covering those specific services (yet).

```
Relying on consent as the legal justification for releasing attributes in all or most cases is not recommended because the person may not be able to provide (legally) valid consent, e.g. when the person needs to access a service to perform her tasks/duties as a researcher or student. Consent should only be used as a "last resort" to help enable access in absence of other rules/policies.
```

For specifics on consent configuration for now please see the Shibboleth IDP documentation on consent and discuss on the community mailing list!

If you're done with editing your filter configuration you can activate the changes in a running IDP by reload only the IDP's attribute filter sub-system, without having to restart the whole Java container:

```
/opt/shibboleth-idp/bin/reload-service.sh -id shibboleth.AttributeFilterService
```

Check your /opt/shibboleth-idp/logs/idp-process.log for any ERROR or WARN occurrences.