

Configure a Linux Server's Attribute Resolution

Once the Moonshot Libraries have been installed on a Server and they have been configured to connect to a local Moonshot RP Proxy, they need to be configured to resolve attributes received in the Moonshot SAML assertion or RADIUS attributes to something the local service can do something with.

Contents

- [Introduction](#)
- [Configure attribute resolution using the Shibboleth SP library](#)
 - [Mapping to an account specified in a SAML attribute](#)
 - [Mapping to an account specified in a RADIUS attribute](#)
- [Configure attribute resolution using internal JSON attribute resolution](#)
 - [Create the JSON mapping file](#)
 - [attribute_operation and attribute_source](#)
 - [Mapping to an account specified in a SAML attribute](#)
 - [Mapping to an account specified in a RADIUS attribute](#)

Introduction

The Moonshot GSS mechanism exports the attributes received in the SAML Assertion (see [Issue SAML Assertions](#)) and RADIUS Access-Accept messages as GSS Name Attributes, as described in [RFC 7056](#).

While some applications might be able to consume these Name Attributes directly, some others might require that a well-known attribute is sourced, so they can consume it without having to understand every possible name. An example of the latter is OpenSSH, where an attribute called `local-login-user` is checked to obtain the name of the account the authenticated user is authorised to access.

The Moonshot software provides two different ways to perform attribute mapping (that is, the generation of an attribute using another attribute's value). The first one uses (included in the `moonshot-gss-eap` package) the Shibboleth SP library, and provides a set of powerful transformation primitives. However, this comes at the cost of adding additional dependencies and configuration complexity. The second one (included in the `moonshot-gss-eap-noshib`) uses a simple JSON file to define very simple mapping rules.



The default configuration for attribute resolution uses the Shibboleth SP library. On certain platforms, attribute resolution is not available with Shibboleth and attribute resolution with JSON must be used. This is usually the case on platforms where Shibboleth is not available or too heavy-weight.

Configure attribute resolution using the Shibboleth SP library

Moonshot by default uses [Shibboleth](#) libraries to parse RADIUS and SAML attributes.

SAML assertions can be embedded inside RADIUS responses by the IdP or the RP Proxy, allowing an IdP or RP Proxy to exercise a very fine-grained authorisation policy. One potential use of this is to allow the Moonshot IdP or RP Proxy to specify which account the user should log in to your service as. RADIUS attributes, such as the `User-Name` attribute, are simply mapped with a special type of Shibboleth attribute. To do this, enable the functionality in Shibboleth as follows.

Edit `/etc/shibboleth/shibboleth2.xml` and modify the lines after the opening `<SPConfig ... clockSkew="180">` stanza:



Shibboleth 2.x only

Insert these lines immediately after the opening stanza:

```
<OutOfProcess tranLogFormat="%u|%s|%IDP|%i|%ac|%t|%attr|%n|%b|%E|%S|%SS|%L|%UA|%a">
  <Extensions>
    <Library path="plugins.so" fatal="true" />
  </Extensions>
</OutOfProcess>
```



Shibboleth 3.x only

Modify the `OutOfProcess` stanza as follows:

```
<OutOfProcess tranLogFormat="%u|%s|%IDP|%i|%ac|%t|%attr|%n|%b|%E|%S|%SS|%L|%UA|%a">
  <Extensions>
    <Library path="plugins.so" fatal="true" />
  </Extensions>
</OutOfProcess>
```

Mapping to an account specified in a SAML attribute

To map an attribute in a SAML assertion embedded in a RADIUS response, you should do the following:



Example

In this example, the service maps a SAML attribute to a local user account (via `local-login-user`)

1. Edit `/etc/shibboleth/attribute-map.xml` and find the SAML attribute that the Moonshot IdP or RP Proxy will be sending you that contains the username.



Example

We want to map from the incoming SAML2 representation of "eduPersonEntitlement"

```
<Attribute name="urn:oid:1.3.6.1.4.1.5923.1.1.1.7" id="entitlement"/>
```

2. Change the id of the attribute from "entitlement" to "local-login-user".



Example

We change the attribute defining the SAML2 representation of "eduPersonEntitlement" such that its ID becomes "local-login-user"

```
<Attribute name="urn:oid:1.3.6.1.4.1.5923.1.1.1.7" id="local-login-user"/>
```

Mapping to an account specified in a RADIUS attribute

To parse a RADIUS attribute (such as the `User-Name` attribute), you should do the following:



Example

In this example, the service maps the `User-Name` attribute to a local user account (via `local-login-user`)

1. Edit `/etc/shibboleth/shibboleth2.xml` and find the line `<AttributeExtractor type="XML" ...>` further down in the file, duplicate it and modify the duplicate as follows:

```
<AttributeExtractor type="GSSAPI" validate="true" reloadChanges="false" path="attribute-map.xml"/>
```



You may want to store your GSSAPI attributes in a separate file. In this case, amend the path above to the new file.

2. Edit `/etc/shibboleth/attribute-map.xml` and find the first attribute line that does not use an `AttributeDecoder`. If you store your GSSAPI attributes in a separate file, modify that file instead.

Example

We will duplicate the incoming SAML2 representation of "eduPersonEntitlement"

```
<Attribute name="urn:oid:1.3.6.1.4.1.5923.1.1.1.7" id="entitlement"/>
```

3. Duplicate the line. The name format for GSSAPI attributes is somewhat different. It does not use the OID namespace; instead it uses the IETF namespace. The attribute name is also different.

Example

We change the attribute defining the SAML2 representation of "eduPersonEntitlement" such that its ID becomes "local-login-user" and it uses a different namespace

```
<GSSAPIAttribute name="urn:ietf:params:gss:radius-attribute 1" id="local-login-user"/>
```



The numeral 1 in the name of the attribute refers to RADIUS attribute 1, which is the User-Name. A more extensive compendium of attribute numbers is available at [IANA's number registry](#).

Configure attribute resolution using internal JSON attribute resolution



Internal JSON resolution is only available in the `moonshot-gss-eap-noshib` package. This package is the same as the classic `moonshot-gss-eap` package, but is built without Shibboleth support.

Moonshot now also supports the use of a JSON file that performs basic mapping of attributes in the Moonshot response to local attributes as needed. The most basic functionality will simply copy the value from an attribute provided, but the built-in function also allows the setting of some values statically.

The difference between the internal JSON attribute resolution and the Shibboleth-based resolution is that the internal JSON resolver does not use any namespace by default at all and will require the URN (the identifier) for the attribute to be specified in the `urn:ietf:gss:*` namespace, while the Shibboleth-based resolution defaults to the SAML namespaces and requires that only RADIUS-based attributes be specified with the `urn:ietf:gss:*` namespace.

Create the JSON mapping file

1. Create the file `/etc/moonshot/local-attributes.json`.
2. Create a JSON block that contains something like this:

```
{
  "target_attribute": {
    "attribute_operation": "attribute_source"
  },
  "an_other_attribute": {
    "attribute_operation": "other_attribute_source"
  }
}
```

attribute_operation and attribute_source

The `attribute_operation` item in the above JSON snippet takes one of two values, `values` or `copy_from`

1. `values`: Used when defining fixed strings or values. In this case the `attribute_source` item must be an array of strings

✔ Example

```
{
  "my_multi_value_attribute": {
    "values": ["this is value 1", "this is value 2"]
  }
}
```

2. `copy_from`: Used when copying values from either SAML or RADIUS attributes in the Moonshot reply. The `attribute_source` item will be either a string that contains the OID-based name of the attribute, or an array of strings denoting multiple attributes in the Moonshot reply in order of precedence

✔ Example

Copying a single attribute value from the RADIUS `User-Name` attribute:

```
{
  "user_name": {
    "copy_from": "urn:ietf:params:gss:radius-attribute 1"
  }
}
```

Copying an attribute value from a series of attributes in declining precedence (RADIUS `User-Name` first, then the SAML attribute `uid`):

```
{
  "user_name": {
    "copy_from": ["urn:ietf:params:gss:radius-attribute 1",
                  "urn:ietf:params:gss:federated-saml-attribute urn:oasis:names:tc:SAML:2.0:attrname-format:uri urn:oid:0.9.2342.19200300.100.1.1"]
  }
}
```

Mapping to an account specified in a SAML attribute

To map an attribute in a SAML assertion embedded in a RADIUS response, you should do the following:

✔ Example

In this example, the service maps a SAML attribute to a local user account (via `local-login-user`)

1. Edit `/etc/moonshot/local-attributes.json` as follows to define the SAML attribute that the Moonshot IdP or RP Proxy will be sending you that contains the username.

✔ Example

We want to map from the incoming SAML2 representation of "eduPersonEntitlement"

```
{
  "local-login-user": {
    "copy_from": "urn:ietf:params:gss:federated-saml-attribute urn:oasis:names:tc:SAML:2.0:attrname-format:uri urn:oid:1.3.6.1.4.1.5923.1.1.1.7"
  }
}
```

Mapping to an account specified in a RADIUS attribute

To parse a RADIUS attribute (such as the `User-Name` attribute), you should do the following:

Example

In this example, the service maps the `User-Name` attribute to a local user account (via `local-login-user`)

1. Edit `/etc/moonshot/local-attributes.json` as follows to define the RADIUS attribute that the Moonshot IdP or RP Proxy will be sending you that contains the username.

Example

We want to map from the incoming RADIUS representation of `User-Name` (the standard way to send a user name):

```
{
  "local-login-user": {
    "copy_from": "urn:ietf:params:gss:radius-attribute 1"
  }
}
```



The numeral 1 in the name of the attribute refers to RADIUS attribute 1, which is the `User-Name`. A more extensive compendium of attribute numbers is available at [IANA's number registry](#).