SmartCloud

IBM

Federated Identity & Federated Service Provider Support for OpenStack Clouds

Joe Savak - Rackspace Identity Product Manager
Brad Topol - IBM Distinguished Engineer
Jorge Williams - Rackspace Integration Architect
Steve Martinelli - IBM Software Developer (Keystone Core)

What was delivered in Icehouse?

Officially, from the Icehouse release notes:

> "The OS-FEDERATION extension allows Keystone to consume federated authentication via an Apache module for multiple Identity Providers, and mapping federated attributes into OpenStack group-based role assignments"
> – Dolph Mathews (Keystone PTL)

Outline:

- New Keystone OS-FEDERATION APIs
  - Identity Providers
  - Protocols
  - Mappings
- Mappings
  - Motivation for Mappings
  - Setting up OpenStack Groups
  - SAML Assertions
  - Creating a Mapping
- Authenticating
  - Motivation for Authenticating
  - Authenticating with Keystone – Part 1
  - Authenticating with Keystone – Part 2
  - Authenticating with Keystone – Part 3

New Keystone OS-FEDERATION APIs

Identity Providers: `/OS-FEDERATION/identity_providers/{idp_id}`

- An Identity Provider is a third party service that is trusted by the Identity API to authenticate identities.
- Register SAML Identity Providers such as ADFS or Tivoli Federated Identity Manager.

```
{
    "identity_provider": {
        "description": "Stores ACME identities",
        "enabled": true,
        "id": "ACME",
    }
}
```

- description (string) - Describes the identity provider.
- enabled (boolean) - Indicates whether this identity provider should accept federated authentication requests.
- id (string) - User-defined unique id to identify the identity provider.

New Keystone OS-FEDERATION APIs

Protocols:

`/OS-FEDERATION/identity_providers/{idp_id}/protocols/{protocol_id}`

- A Protocol entry contains information that dictates which mapping rules to use for a given incoming request. An IdP may have multiple supported protocols.
- Currently, only the SAML 2.0 federation protocol is supported. However, the framework is extensible to support other federation protocols, i.e.: OpenID, WS-Federation, SAML 1.0.
- Identity Providers can communicate in many protocols, so associate an ***Identity Provider*** with a ***Mapping***, based on a protocol.

```
{
    "protocol": {
        "id": "saml2",
        "mapping_id": "xyz234",
    }
}
```

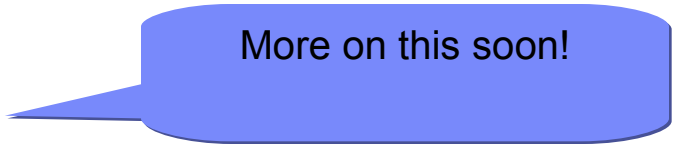
- mapping_id (string) - Indicates which mapping should be used to process federated authentication requests.
- id (string) - User-defined unique id to identify the protocol.

New Keystone OS-FEDERATION APIs

Mappings: `/OS-FEDERATION/mappings/{mapping_id}`

- A mapping is a set of rules to map federation protocol attributes to Identity API objects. An Identity Provider can have a single mapping specified per Protocol. A Mapping is simply a list of rules.
- A mapping is a method to translate remote attributes (from an Identity Provider) to local attributes (Keystone entities).
- Mappings are created as a top level resource so as to enable re-use between Identity Providers.

```
{
    "mapping": {
        "id": "ACME_MAP",
        "rules": [...],
    }
}
```



- rules (list) - Each object contains a rule for mapping attributes to Identity API concepts. A rule contains a remote attribute description and the destination local attribute.
- id (string) - User-defined unique id to identify the mapping.

Motivation for Mapping

Setting the scene:

- In Keystone, **authentication** is performed with password or token.
- **Authorization** is performed by ensuring a **user or group**, has a **role**, on a **project or domain**.
  - i.e., does requesting user **btopol**, have role **developer**, on project **services**?
  - or does requesting user belong to **group** that has role **developer**, on project **services**?

Identifying the problem:

- Users exist on an Identity Provider, not in Keystone.
- Thus, **authenticating** nor **authorizing** a user can be performed locally.
- An Identity Provider will only return Identity attributes (such as user and group information), not authorization attributes.
- Assuming an Apache module can handle the **authentication** between Keystone and the IdP, we still don't have a solution for **authorization.**

Finding a solution:

- Create a mapping to handle **authorization**, it can establish relationships between Keystone entities and Identity Provider attributes.
- Create a 1:1 relationship between Identity Provider groups and Keystone groups
  - Only create groups that will be authorized to perform federated authentication.
- Have a user authenticate with an Identity Provider, and be mapped to a Keystone group, once mapped, the user will **inherit** the roles from the group.

Setting up OpenStack for Groups

- Create groups, that have a role(s) on a project or domain.
- Can be done via CLI for convenience.

```
openstack@plwdevstack:/opt/stack/devstack$ openstack group create regular_employees_canada
+-------------+----------------------------------------------------------------------------------+
| Field       | Value                                                                            |
+-------------+----------------------------------------------------------------------------------+
| description |                                                                                  |
| domain_id   | default                                                                          |
| id          | af27bac827014e67888a40c53015f4dc                                                 |
| links       | {u'self': u'http://10.0.2.15:5000/v3/groups/af                              c'} |
| name        | regular_employees_canada                                                         |
+-------------+----------------------------------------------------------------------------------+
openstack@plwdevstack:/opt/stack/devstack$ openstack group create swg_canada
+-------------+----------------------------------------------------------------------------------+
| Field       | Value                                                                            |
+-------------+----------------------------------------------------------------------------------+
| description |                                                                                  |
| domain_id   | default                                                                          |
| id          | 8ca506c53607452cb22b7e8914ad0214                                                 |
| links       | {u'self': u'http://10.0.2.15:5000/v3/groups/8ca506c53607452cb22b7e8914ad0214'} |
| name        | swg_canada                                                                       |
+-------------+----------------------------------------------------------------------------------+
openstack@plwdevstack:/opt/stack/devstack$ openstack role list
+----------------------------------+---------------+
| ID                               | Name          |
+----------------------------------+---------------+
| 050d34ad50b143d5a376f96b01ac2d19 | Member        |
| 223f5e3b63fb4e57aa50b89616d9f1bb | ResellerAdmin |
| 321470e2e289410e9cbd6db42145fe81 | admin         |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_      |
| ca7237dafee14673a6229b1d95a56e8d | service       |
| e09fe5d2fcd44b11840ebf1231310081 | anotherrole   |
+----------------------------------+---------------+
openstack@plwdevstack:/opt/stack/devstack$ openstack project list
+----------------------------------+--------------------+
| ID                               | Name               |
+----------------------------------+--------------------+
| 2f26be3e34b047d782590e62b0f3cd29 | demo               |
| b9b23d0b341e4338a4d76ad09c1b2dd8 | service            |
| ca53b4510a4146e38d31f8f3957d5ded | admin              |
| fef157813a8e4b50a98f501d2e76d84c | invisible_to_admin |
+----------------------------------+--------------------+
```

Keystone groups have a globally unique id

Keystone Group IDs:

- regular_employees_canada
  - (af27ba … 15f4dc)
- swg_canada
  - (8ca506 … ad0214)

Associate **groups** with a **role** on a **project**.

(Roles and projects listed on left side)

```
$ openstack role add service --project service --group swg_canada
$ openstack role add Member --project service --group swg_canada
$ openstack role add admin --project service --group regular_employees_canada
$ openstack role add Member --project service --group regular_employees_canada
```

SAML Assertions

- A snippet from a SAML assertion is seen below, and has the necessary user and group information (from an IdP perspective).
- The 'roles' in the SAML attributes are the IdPs method of assigning groups, these need to be mapped back to the groups that were created in the previous step.

```
<saml:AttributeStatement>
       <saml:Attribute Name="Role">
           <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
                                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                                xsi:type="xs:string"
                                >IBM Regular Employees Canada</saml:AttributeValue>
       </saml:Attribute>
       <saml:Attribute Name="Role">
           <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
                                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                                xsi:type="xs:string"
                                >SWG Canada</saml:AttributeValue>
       </saml:Attribute>
  </saml:AttributeStatement>
```

IdP Group attributes:

- IBM Regular Employees Canada
- SWG Canada

Keystone Group IDs:

- regular_employees_canada
  - (af27ba … 15f4dc)
- swg_canada
  - (8ca506 … ad0214)

Adding the Mapping

- Create a mapping to map the IdP attributes to Keystone entities.
- Example request body sent to `OS-FEDERATION/mappings/BP_MAP`

```
{
  "mapping": {
    "rules": [
    {
      "local": [
      {
        "group": {
          "id": "af27bac827014e67888a40c53015f4dc"
          }
        }
      ],
      "remote": [
      {
        "type": "Role",
        "any_one_of": [
          "IBM Regular Employees Canada"
        ]
      }
      ]
    }
    ]
  }
}
```

In this case 'IBM Regular Employees Canada' maps to ID 'af27ba … 15f4dc'

```
"rules": [
{
  "local": [{
    "user": {
      "name": "{0}"
    }}],
    "remote": [{
      "type": "sub"
    }]
},
{
  "local": [{
    "group": {
      "id": "8ca506c53607452cb22b7e8914ad0214"
  }}],
  "remote": [{
    "type": "Role",
    "any_one_of": [
      "SWG Canada"
    ]}]
  }
]
```

A mapping can have many rules!

Keystone Group IDs:

- regular_employees_canada
  - (af27ba … 15f4dc)
- swg_canada
  - (8ca506 … ad0214)

Mapped!

IdP Group attributes:

- IBM Regular Employees Canada
- SWG Canada

Motivation for Authenticating

Setting the scene:

- Scope is defined as a resource the user wishes to access.
- In Keystone, authentication may result in a scoped or unscoped token.
  - Depends on the request body, if a scope is provided.
- Authorization depends on, if the authenticated user is authorized to access the resource (based on his/her roles) defined in the scope.
  - Just because a user has authenticated, does not mean he is authorized!

Identifying the problem:

- The new user doesn't know what resources (projects/domains) he/she has access to.

Finding a solution:

- Create new APIs to allow a look-up, based on group Ids.
- Perform authentication and authorization in a few steps.
  - Initially retrieve an unscoped token.
  - Look up which resources the group has access to.
  - Retrieve a scoped token.
- Success!

Authenticating with Keystone - Part 1

Request an unscoped token:

`/OS-FEDERATION/identity_providers/{idp_id}/protocols/{protocol}/auth`

- A federated user may request an unscoped token, which can be used to get a scoped token.
- Supports both Web Single Sign-On (WebSSO) and Enhanced Client Proxy (ECP) workflows.
- The returned Token ID is contained in the response header and the Token Data will contain information about the groups to which the federated user belongs.

Response Header: `X-Auth-Token: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`

Response Body:

```
{
    "token": {
        "methods": [
            "saml2"
        ],
        "user": {
            "id": "joeuser%40ca.ibm.com",
            "name": "joeuser@ca.ibm.com",
            "OS-FEDERATION": {
                "identity_provider": "BP",
                "protocol": "SAML",
                "groups": [
                    {"id": "af27ba … 15f4dc"},
                    {"id": "8ca506 … ad0214"}
                ]
            }
        }
    }
}
```

Standard Keystone token ID (PKI, UUID …)

Data from the SAML assertion

Data from the IdP and protocol associated with the Mapping

Data from the mapping

Authenticating with Keystone - Part 2

Look up authorized resources:

- A user may be authorized to access either a project or domain.

- List projects a federated user can access:
  - `GET /OS-FEDERATION/projects`
  - Returns a collection of projects to which the federated user has authorization to access.
  - To access this resource, an unscoped token is used, the user can then select a project and request a scoped token.
  - Only enabled projects will be returned.

- List domains a federated user can access:
  - `GET /OS-FEDERATION/domains`
  - Returns a collection of domains to which the federated user has authorization to access.
  - To access this resource, an unscoped token is used, the user can then select a domain and request a scoped token.
  - Only enabled domains will be returned.

Authenticating with Keystone - Part 3

Request a scoped token: `/auth/tokens`

- Once a user knows the project or domain id, a request can be made to retrieve a token that has access to that project or domain.
- The returned token shows the roles the user has inherited, as well as the project or domain that was requested.

Request Body:

```
{
  "auth": {
    "identity": {
      "methods": [
        "saml2"
      ],
      "saml2": {
        "id":
"wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
      }
    }
  },
  "scope": {
    "project": {
      "id": "263fd9"
    }
  }
}
```

Taken from the response header in Step 1

Response Body:

Our roles on the project! SUCCESS!

```
{
  "token": {
    "methods": ["oidc"],
    "roles": [
      { "id": "050d34ad50b143d5a376f96b01ac2d19",
        "name": "Member" },
      { "id": "ca7237dafee14673a6229b1d95a56e8d",
        "name": "service" }
    ],
    "expires_at": "2014-03-28T03:07:42.027427Z",
    "project": {
      "domain": {
        "id": "default",
        "name": "Default"
      },
      "id": "b9b23d0b341e4338a4d76ad09c1b2dd8",
      "name": "service"
    },
    "user": {
      "id": "joeuser%40ca.ibm.com",
      "name": "joeuser@ca.ibm.com"
    },
    "issued_at": "2014-03-28T02:07:42.027492Z"
  }
}
```

Questions?

Rethink IT.
Reinvent Business.