Multi-tenant for ONAP

Beyuan Zhang
zhangbeiyuan@huawei.com
Seshu Kumar M
Seshu.Kumar.M@huawei.com
Multi-tenant for ONAP

After login, share the current user to ONAP

ONAPaaS Portal

Tenants

Portal

ONAP

Portal

（New）Multi-tenant-service:

- DB: relations of tenant and building block
- API: filter block by tenantID

SDC  VID  policy  …other

SDC  VID  policy  A&AI

Merit

| The change is thin and controllable, which does not affect the original functions. |

Flaws

| Basic data is not isolated and may conflict. For example, if a tenant creates a block with the same name, it will be identified as the same name in the ONAP system. |

Change 1: The Multi-tenant-service records the home tenants of all data and provides interfaces for adding, deleting, modifying, and querying data.

Change 2: When data is accessed on all pages, data is obtained based on the original process and then filtered based on tenantID.

Change 3: When data is added or deleted, the Multi-tenant-service interface needs to be invoked to update the relationship between data and tenants.
Key Point:

1. Add the tenant_id to session, let ONAP to know who is login.
2. Add a table to record the relationship of tenant_id and buildingblock_id, so 2 step should be done:
   - step1: when create a buildingblock, you should record the relationship into the table;
   - step2: when you want to get the buildingblocks from interface, and then use the relationships to filter the data what you got from DB

We used SDC and AAI to verify the scheme
On SDC: Two points should be done

1. **Simulate tenant user login:**
   In the login interface of the simulate, add tenant to the cookie, and need to write the session in the future.

2. **Insert test API**
   Find the query data interface in the SDC code, and filter the data according to the tenant-Id in the cookie, and need to be transplanted into the multi-tenant component in the future.

   - The relationship between Element and tenant is temporarily saved in memory and needs to be written to the database in the future.

---

**Insert test code in two positions**

- After login, share the current user to ONAP.
- **New** Multi-tenant-service:
  - DB: relations of tenant and building block
  - API: filter block by tenant ID

---

**Current diagram highlights:**

- **tenant1** login
- **Query data**
- ONAPaaS Portal
- Tenants
- Portal
- SDC, VID, policy...
- other
On SDC: Insert tenant_id with simulation

SDC can simulate the login by simulation, modify the original login interface, add the tenant attribute and save it to the cookie.

Login by Tenant Id + User Id: Could operate the data of the tenant only
User Id: cs0008 or jm0007
Password: 123123a

Login by User Id: Could operate all of data
Step 1: Find the API: rest/v1/followed
(Use Chrome to open the SDC page, and open developer-model).

Step 2: To find the code of the API from SDC code by the path.
rest/v1/followed that code is catalog-be/ElementServlet.java

Step 3: Insert this code to filter the result (detail in next page)
SDC Code modification to achieve data filtering

GenericServlet. getFollowedResourcesServices()

```
if (either.left().value != null) {
  eitherLeftData = either.left().value;
} else if (followedResourcesServices instanceof Component) {
  followedResourcesServices = getComponentList(request, followedResourcesServices);
}
```

MultiTenantService.filterResultByTenantId()

```
if (resources.isRight()) {
  return resources;
}
```

// TODO test start: To filter resource by tenant, and tenant id is saved in session.
followedResourcesServices = filterElementByTenantId(resquest, followedResourcesServices);
// TODO test end.
Object data = RepresentationHills.toRepresentation(followedResourcesServices.left().value());
res = buildResponse(getComponentsUtil(), data);
else {
  buildErrorResponse(followedResourcesServices.right().value());
}

// TODO test start: when get data from DB, to filter by tenant id
protected Either<
  Map<String, List>> filterElementByTenantId(String tenantId)
```

```
if (tenantId == null || tenantId.isEmpty()) {
  return resources;
}
```

// if no tenant id, that will return all of data
return resources;

MultiTenantService multisever = new MultiTenantService();
Either<Map<String, List>> filtered = multisever.filterResultByTenantId(resources, tenantId);
return filtered;

Filter data by tenant id
Test Result: Using two tenant-id to create services

Login with **tenant01** + cs0008:

<table>
<thead>
<tr>
<th>Login</th>
<th>Create service: service-Tenant01-001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tenant id:</td>
<td>tenant01 ▼</td>
</tr>
<tr>
<td>User id:</td>
<td>cs0008</td>
</tr>
<tr>
<td>Password:</td>
<td>******</td>
</tr>
</tbody>
</table>

Login with **tenant02** + cs0008:

<table>
<thead>
<tr>
<th>Login</th>
<th>Create service: service-Tenant02-001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tenant id:</td>
<td>tenant02 ▼</td>
</tr>
<tr>
<td>User id:</td>
<td>cs0008</td>
</tr>
<tr>
<td>Password:</td>
<td>******</td>
</tr>
</tbody>
</table>

Login with cs0008:

Quick links:

<table>
<thead>
<tr>
<th>Full Name</th>
<th>User id</th>
<th>Role</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jimmy Hendra</td>
<td>j0003</td>
<td>Admin</td>
<td>create</td>
</tr>
<tr>
<td>Carlos Santos</td>
<td>c0006</td>
<td>Designer</td>
<td>create</td>
</tr>
<tr>
<td>John Deto</td>
<td>j0007</td>
<td>Tester</td>
<td>create</td>
</tr>
<tr>
<td>Andrea Flanders</td>
<td>a0001</td>
<td>Ops</td>
<td>create</td>
</tr>
<tr>
<td>David Shadrin</td>
<td>d0001</td>
<td>Governor</td>
<td>create</td>
</tr>
</tbody>
</table>

So, SDC is easy to do, but AAI is very difficult to change, To be continue…
AAI code analysis --- resource relationship

The main code is in aai-resource:
- aai-oxm-v16.xml: define all API interfaces
- Dbedgerules v16.json: define the relationship between nodes
AAI - @get interface analysis

Interface entry

To query and filter data:

The result returned by this interface is a response structure, and the actual return value is a JSON or XML result in string format. But because we don't know the specific object in here, we can't automatically identify the only key value, so we can't filter the data very well.

Get tenant_id from header:
Summary

1. This is just a simulated multi-tenant scheme, because it does not achieve real data isolation
2. SDC can be implemented through simple changes, but AAI is very difficult to do this, requiring major changes or other better solutions
3. Other modules have yet to be verified
4. There are defects in the scheme itself, such as the inability to create a buildingBlock with the same name, but the single user cannot perceive the error
1. Does the community has plans to implement a multi-tenant solution? And how to do it
2. Welcome everyone to join us to explore a feasible multi-tenant implementation
3. We hope that more people in the community can join us, and discuss how to achieve multi-tenancy.