Microservices Bus Tutorial
Huabing Zhao
Agenda

- MSB Overview
- Service Registration
- Service Discovery
- Example & Demo
- How to integrate with MSB in Amsterdam
MSB Overview-Introduction

MSB (Microservices Bus) provide a comprehensive, end to end solution to support ONAP microservice architecture including service registration/discovery, external gateway, internal gateway, client SDK. It's a pluggable architecture so it can integrate with auth service provider to provide centralized Authentication & Authorization. MSB also provides a service portal to manage the REST APIs.

MSB doesn’t depend on a specific environment. It can work in bare metal, virtual machine or containerized environment.
MSB Overview-Components

- Registry
  Service information storage, MSB leverage Consul as service registry.
- MSB Discovery
  Provides REST APIs for service discovery and registration
- Service Gateway
  Provide service request routing, load balancing and centralized Auth. It can be deployed as external Gateway or Internal Gateway.
- MSB SDK
  Java SDK for point to point communication
Service Endpoint Information Model

```json
{
    "serviceName": "catalog",
    "version": "v1",
    "url": "/api/catalog/v1",
    "protocol": "REST",
    "visualRange": "1",
    "lb_policy": "ip_hash",
    "nodes": [
        {
            "ip": "10.74.55.66",
            "port": "6666",
            "ttl": 0
        },
        {
            "ip": "10.74.56.36",
            "port": "8988",
            "ttl": 0
        }
    ]
}
```

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceName</td>
<td>Service Name</td>
</tr>
<tr>
<td>version</td>
<td>Service Version</td>
</tr>
<tr>
<td>url</td>
<td>the actual URL of the service to be registered</td>
</tr>
<tr>
<td>protocol</td>
<td>supported protocols: 'REST', 'UI', 'HTTP', 'TCP'</td>
</tr>
<tr>
<td>visualRange</td>
<td>Visibility of the service. External (can be accessed by external systems): 0</td>
</tr>
<tr>
<td></td>
<td>Internal (can only be accessed by ONAP microservices): 1</td>
</tr>
<tr>
<td>path</td>
<td>The customized publish path of this service. If path parameter is specified</td>
</tr>
<tr>
<td></td>
<td>when registering the service, the service will be published to api gateway</td>
</tr>
<tr>
<td></td>
<td>under this path. Otherwise, the service will be published to api gateway</td>
</tr>
<tr>
<td></td>
<td>using a fixed format: api/(serviceName) /{version}. The customized publish</td>
</tr>
<tr>
<td></td>
<td>path should only be used for back-compatible.</td>
</tr>
<tr>
<td>lb_policy</td>
<td>Load balancing method, Currently two LB methods are supported, round-robin</td>
</tr>
<tr>
<td></td>
<td>and ip-hash.</td>
</tr>
<tr>
<td>enable_ssl</td>
<td>True if the registered service is based on https. False if the</td>
</tr>
<tr>
<td></td>
<td>registered service is based on http.</td>
</tr>
<tr>
<td>nodes</td>
<td>ip: the ip of the service instance node port: the port of the service</td>
</tr>
<tr>
<td></td>
<td>instance node ttl: time to live, this parameter is reserved for later use</td>
</tr>
</tbody>
</table>
Service Registration – RESTful API

http method: POST
url: http://{msb_ip}:{msb_port}/api/microservices/v1/services

Example:
curl -X POST \  
- H "Content-Type: application/json" \  
- d '{"serviceName": "test", "version": "v1", "url": "/", "protocol": "REST", "lb_policy": "round-robin", "nodes": [ {"ip": "127.0.0.1", "port": "9090"}]}' \  "http://127.0.0.1:10081/api/microservices/v1/services"
Service Registration-OOM Registrar

OOM Registrar can register service endpoints for the microservices deployed by OOM

- OOM deploy/start/stop ONAP components.
- Registrator watches the Kubernetes pod event.
- Registrator registers service endpoint info to MSB. It also updates the service info to MSB when ONAP components are stopped/restarted/scaled by OOM
Use Kubernetes annotations to attach service endpoint metadata to objects.
Service endpoint can be defined at Pod level or Service level
Pod level: leverage the LB capabilities of MSB to distribute requests to multiple pods
Service level: MSB send the request to service(Cluster IP), K8s dispatch the request to the backend Pod

```yaml
apiVersion: v1
kind: Service
metadata:
  name: aai-service
  annotations:
    msb.onap.org/service-info: |
      |
      |
      { |
        "serviceName": "aai-cloudInfrastructure",
        "version": "v1",
        "url": "/cloud-infrastructure",
        "protocol": "REST",
        "lb_policy": "round-robin",
        "visualRange": "1",
        "enable_ssl": "false"
    }
```
Service Registration-MSB SDK

Microservices can use SDK to register themselves to MSB.

```java
public void registerMsb() throws Exception {

    //For real use case, MSB IP and Port should come from configuration
    //file instead of hard code here
    String MSB_IP="127.0.0.1";
    int MSB_Port=10081;

    MicroServiceInfo msinfo = new MicroServiceInfo();

    msinfo.setServiceName("animals");
    msinfo.setVersion("v1");
    msinfo.setUrl("/api/rpc/v1");
    msinfo.setProtocol("REST");
    msinfo.setVisualRange("1");

    Set<Node> nodes = new HashSet<>();
    Node node1 = new Node();
    node1.setIp(InetAddress.getLocalHost().getHostAddress());
    node1.setPort("9090");
    nodes.add(node1);
    msinfo.setNodes(nodes);

    MSBServiceClient msbClient = new MSBServiceClient(MSB_IP, MSB_Port);
    msbClient.registerMicroServiceInfo(msinfo, false);
}
```
Service Discovery-Gateway

- **External service gateway**
  - Expose the services (Rest API, UI pages, etc.) which need to be accessed by external systems
  - Solve the cross-domain issue for web app
  - Protocol transformation/translation between external requests and internal service
  - Centralized Auth

- **Internal API gateway**
  - Routing and load balancing of the API calls within the system
  - Minimize the codes modification for service consumer

Both the external and internal gateway can be deployed as a cluster (multiple instances) to avoid single point of failure
Service Discovery-MSB SDK

Microservices can use SDK to MSB SDK to discovery and access other microservices within ONAP.

```java
public static void main(String[] args) throws IOException {
    // For real use case, MSB IP and Port should come from configuration
    // file instead of hard code here
    String MSB_IP = "127.0.0.1";
    int MSB_Port = 10081;

    MSBServiceClient msbClient = new MSBServiceClient(MSB_IP, MSB_Port);

    RestServiceCreator restServiceCreator =
        new RestServiceCreator(msbClient);

    AnimalServiceClient implProxy =
        restServiceCreator.createService(AnimalServiceClient.class);

    Animal animal = implProxy.queryAnimal("panda").execute().body();
    System.out.println("animal: " + animal);
}
Example & Demo

- **Start MSB services**
  1. Run the Consul dockers.
     ```
     sudo docker run -d --net=host --name msb_consul consul agent -dev
     ```
  2. Run the MSB dockers.
     Login the ONAP docker registry first: docker login -u docker -p docker nexus3.onap.org:10001
     ```
     sudo docker run -d --net=host --name msb_discovery nexus3.onap.org:10001/onap/msb/msb_discovery
     ```
     ```
     sudo docker run -d --net=host -e "ROUTE_LABELS=visualRange:1" --name
     msb_internal_apigateway nexus3.onap.org:10001/onap/msb/msb_apigateway
     ```

- **Explore the MSB portal.**
  [http://127.0.0.1/msb](http://127.0.0.1/msb)

- **Use MSB SDK to register/access services**
  [https://gerrit.onap.org/r/gitweb?p=msb/java-sdk.git;a=tree;f=example;h=1c331f86cbbcdb8cc2935d8ac41169da1a523ec5;hb=refs/heads/master](https://gerrit.onap.org/r/gitweb?p=msb/java-sdk.git;a=tree;f=example;h=1c331f86cbbcdb8cc2935d8ac41169da1a523ec5;hb=refs/heads/master)
How to integrate with MSB in Amsterdam

- Register the service endpoints to the wiki page
- Use annotations to attach service endpoint metadata to Kubernetes pod or service objects
- Use MSB SDK/Internal API Gateway to access services

Useful resources

https://wiki.onap.org/display/DW/ONAP+Services+List
https://wiki.onap.org/display/DW/MSB+Test+Environment+Setup
https://wiki.onap.org/display/DW/Microservice+Bus+API+Documentation