ICN DCM

Motivation

In ICN, we have ONAP4k8s as the service orchestration and Kubernetes as the resource orchestration. In the edge deployment, there will be multiple end-users sharing the same edge compute resource. The challenges are to isolate the end-users deployment and allocate the resource as per their demand and quota. This proposal addresses these challenges by creating a "Logical cloud" for the set of users, and provide logical isolation and resource quota.

Goal(in Scope)

Focusing on the solution within service orchestration

Goal(Out of Scope)

Working in Kubernetes core or API is clearly out of the scope of these documents. There are the solutions available to provide a separate control plane to each tenant in a cluster. But the creation of tenant within a cluster does not address the shared clusters and tenant creation should be at service orchestration instead of resource orchestration

Outline

In this section, we define Logical cloud in general for the Service Orchestration engine. A Logical cloud can be defined as a group of resources bounded and isolated amount of compute, storage, networking and control plane in a kubernetes cluster. A Logical cloud can also be defined as a group of users slicing a bounded resource allocated for them. These resources can be as follows:

- CPU, Memory, Extended Resources
- Network bandwidth, I/O bandwidth, Resource orchestration(Kubernetes) cluster resource
- Resource reservation to provide the Guaranteed QoS in Resource orchestration(Kubernetes)

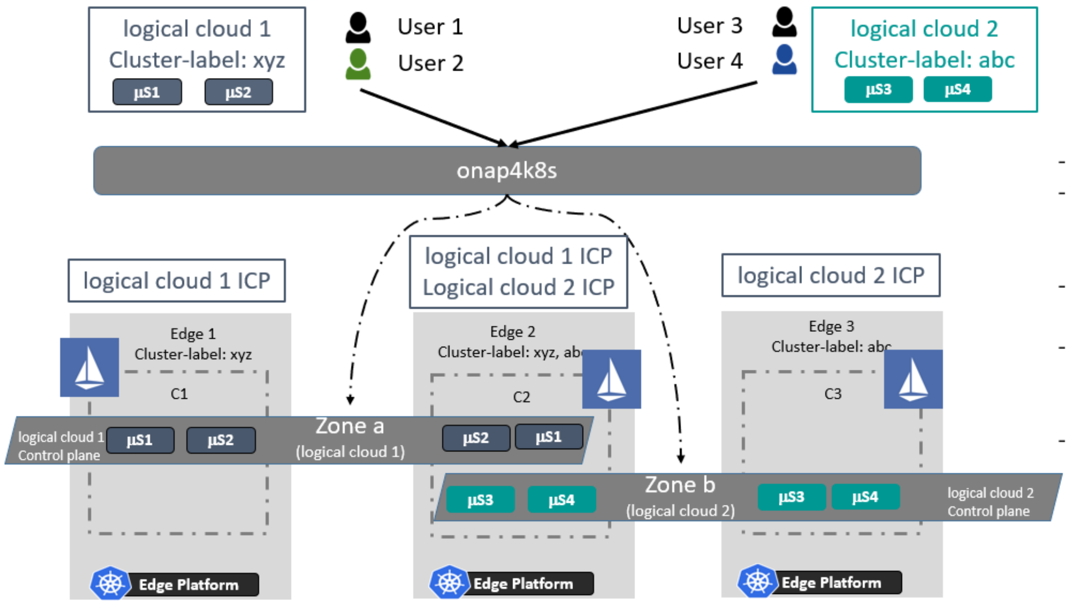
Requirement

1. For a service provider, a Logical cloud is basically a group of end-user sharing the same cluster, we have to make sure that the end-user resources are tracked and accountable for their consumption in a cluster
2. In a few cases, admin or end-user application is shared among multiple Logical cloud, in such case application resource should be tracked across the cluster
3. Centralization resource quota or the allocation limits record should be maintained by admin or for the end-user. For example, just a " Resource query" API to Service Orchestration (ONAP4K8s) should display the resource quota and policy for each end-user or logical cloud
4. In Edge use case, the service orchestration like ICN should get the resource details across multiple clusters by resource orchestration, should set the resource allocation for the cluster and decide the scheduling mechanism
5. User credential centralization with application orchestration

Distributed Cloud Manager

Objectives:

- **User creation**
- **Namespace creation**
- **Logical cloud creation**
- **Resource isolation**

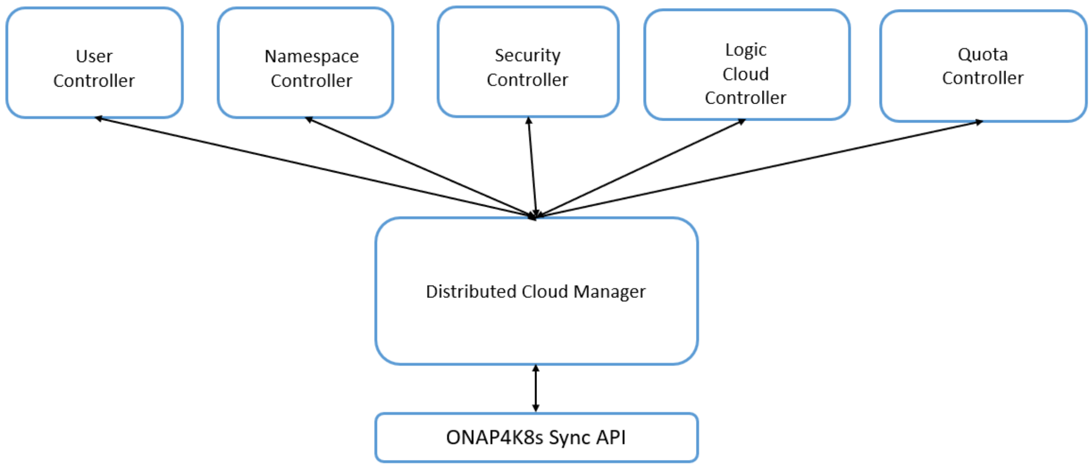


- **User creation**
- **logical cloud creation based on cluster-labels**
- **Istio control plane per logical cloud**
- **Instance creation with respect to logical cloud**
- **Isolate microservice with respect to logical cloud and users**

Assumption:

1. During the Cluster registration to the ONAP4K8s, the ONAP4K8s HPA features associate each cluster with the labels
2. These labels are used by the DCM to identities the cluster and create the logical cloud

DCM Block diagram



DCM Flow:

1. All components are exposed as DCM microservices and queries are made by Rest API
2. DCM User controller microservices API is used to create the users with logical cloud admin and their associated logical namespace in the each cluster using Cluster labels
3. DCM Manager looks for the quota information, if the quota information is not available it will apply the default quota for memory, CPU and kubernetes resources
4. DCM Manager Microservices queries the database to create users, namespace, security controller root CA
5. DCM Manager create logical cloud using Istio control plane using namespace and security controller root CA
6. The quota for the logical cloud could be tuned even after the logical cloud has been created

Logical cloud creation(With default resource quota & users)

The following steps explain how to run the tenant controller in kubernetes

logical cloud creation

```
URL: /v2/projects/<project-name>/logical-clouds
POST BODY:
{
 "name": "logical-cloud-1",   //unique name for the new logical cloud
 "description": "logical cloud for walmart finance department",  //description for the logical cloud
 "cluster-labels": "abc,xyz",
 "resources": {
        "cpu": "400",
        "memory": "1000Gi",
        "pods": "500",
    "dummy/dummyResource": 100,
 },
  "user" : [{
    "name" : "user-1",  //name of user for this cloud
    "type" : "certificate",   //type of authentication credentials used by user (certificate, APIKey, UNPW)
    "certificate" : "/path/to/user1/logical cloud-1-user1.csr" ,  //Path to user certificate
    "permissions" : {
       "apiGroups" : ["stable.example.com"]
       "resources" : ["secrets", "pods"]
       "verbs" : ["get", "watch", "list", "create"]
     },
        "quota" : {
                "cpu": "100",
                "memory": "500Gi",
                "pods": "100",
            "dummy/dummyResource": 20
        }]
  }
}

curl -d @create_logical_cloud-1.json http://onap4k8s:<multicloud-k8s_NODE_PORT>/v2/projects/<project-name>
/logical-clouds \
        --key ./logical cloud-t1-admin-key.pem \
          --cert ./logical cloud-t1-admin.pem \

Return Status: 201
Return Body:
{
  "name" : "logical-cloud-1"
  "user" : "user-1"
  "Message" : "logical cloud and associated user successfully created"
}
```

Creating users:

Logical cloud admin key and certificate should be created by Logical cloud admin(the one who create the curl command). Authentication is required for the curl command. DCM should have the Admin logical cloud information to authenticate the curl command. Unauthorized users can't create the logical cloud.

How the user certificate should be created?

- Create a private key for user1
  - openssl genrsa -out **logical cloud-1-user-1.key** 2048

- Create a certificate sign request logical cloud-1-user1.csr
  - openssl req -new -key logical cloud-1-user-1.key -out **logical cloud-1-user1.csr** -subj "/CN=user-1/O=logical cloud-1“

This information should be created before creating logical cloud and inserted in the logical cloud creation

Binding the user certificates with the cluster?

User controller does the following steps to bind the user certificate with the cluster using the **cluster-labels : abc and xyz.** **Itohan Ukponmwan** **Please get the GET URL from HPA controller to get the cluster list with cluster labels**

DCM queries HPA controller the list of cluster having cluster-labels abc and gets the cluster list c1 and c2

Each cluster(C1, C2) has the Kubernetes cluster certificate cluster (CA – c1-ca.crt & c1-ca.key), generate the final certificate **logical cloud-1-user1.crt** by using logical cloud-1-user1.csr (do the same for the cluster c2). user controller does the following steps once the logical cloud curl command is post through grpc with goclient API

> $ openssl x509 -req -in logical cloud-1-user1.csr -CA CA_LOCATION/c1-ca.crt -CAkey CA_LOCATION/c1-ca.key -CAcreateserial -out **logical cloud-1-user1-c1.crt** -days 500
>
> $ kubectl –kubeconfig=/path/to/c1/kubeconfig config set-credentials **user-1** --client-certificate=./ **logical cloud-1-user1-c1.crt** --client-key=./**logical cloud-1-user-1.key**

The following steps explain how to run the tenant controller in kubernetes

get users

```
GET URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/users
RETURN STATUS: 200
RETURN BODY:
{
  users" : [{
    "name" : "user-1",  //name of user for this cloud
    "type" : "certificate",   //type of authentication credentials used by user (certificate, APIKey, UNPW)
    "certificate" : "/path/to/user1/logical cloud-1-user1.csr" ,  //Path to user certificate
    "permissions" : {
       "apiGroups" : ["stable.example.com"]
       "resources" : ["secrets", "pods"]
       "verbs" : ["get", "watch", "list", "create"]
     },
        "quota" : {
                "cpu": "100",
                "memory": "500Gi",
                "pods": "100",
            "dummy/dummyResource": 20
        }
        },
    {
    "name" : "user-2",  //name of user for this cloud
    "type" : "certificate",   //type of authentication credentials used by user (certificate, APIKey, UNPW)
    "certificate" : "/path/to/user2/logical cloud-1-user1.csr" ,  //Path to user certificate
    "permissions" : {
       "apiGroups" : ["stable.example.com"]
       "resources" : ["secrets", "pods"]
       "verbs" : ["get", "watch", "list", "create"]
     },
        "quota" : {
                "cpu": "100",
                "memory": "500Gi",
                "pods": "100",
            "dummy/dummyResource": 20
        }
        }
]

DELETE
URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/users
URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/user/<user-name>

RETURN STATUS: 204
```

Creating namespaces:

DCM queries the namespace controller through grpc to create namespace **"logical cloud-1-ns"** for the cluster with cluster labels abc and xyz. Namespace controller does the the following steps, to create the namespace and set the user with namespace through grpc with goclient API

> $ kubectl create namespace logical cloud-1-ns --kubeconfig=/path/to/c1/kubeconfig
>
> $ kubectl config set-context logical-cloud-1-user-1-context --cluster=c1 --namespace= logical cloud-1-ns --user=user1 --kubeconfig=/path/to/c1/kubeconfig

The following steps explain how to run the tenant controller in kubernetes

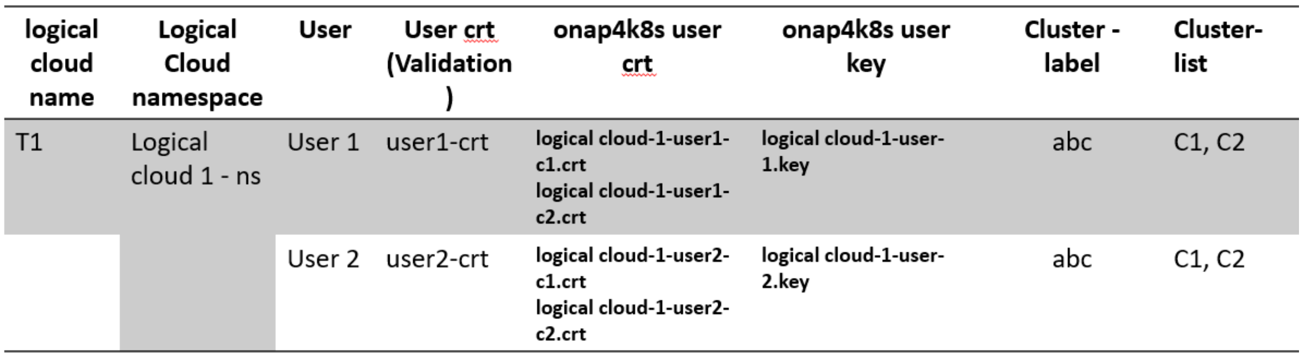
namespace api

```
GET URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/namespaces
RETURN STATUS: 200
RETURN BODY:
{
  "clusters": {c1, c2}
  namespaces" : {
    "name" : "logical cloud-1-ns",  //name of namespace for the logical cloud
        }
}

DELETE
URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/namespaces

RETURN STATUS: 204
```

DCM Database



logical cloud name	Logical Cloud namespace	User	User crt (Validation)	onap4k8s user crt	onap4k8s user key	Cluster - label	Cluster-list
T1	Logical cloud 1 - ns	User 1	user1-crt	**logical cloud-1-user1-c1.crt** **logical cloud-1-user1-c2.crt**	**logical cloud-1-user-1.key**	abc	C1, C2
		User 2	user2-crt	**logical cloud-1-user2-c1.crt** **logical cloud-1-user2-c2.crt**	**logical cloud-1-user-2.key**	abc	C1, C2

DCM Database is based on Mongo DB.

Creating Logical cloud

1. Logical cloud controller uses the Istio to create logical control using the https://istio.io/docs/setup/install/multicluster/gateways/ .
2. DCM manager queries the Security controller with /v1/cadist/projects/{project-name}/logicalclouds/{logicalcloud-name}/clusters/{cluster-name} to get the bundle details for the cluster C1 and C2
3. The expectation that "JSON bundle" should provide the path to the root cert.
4. Logical cloud controller creates Isito control plane in Cluster C1 and C2 for namespace **logical cloud-1-ns-istio-system**

keys api

```
URL: /v2/projects/<project-name>/logical-clouds/control-plane
POST BODY:
{
 "name": "logical-cloud-1",   //unique name for the new logical cloud
 "namespace": "Logical-cloud-1-istio-system",
 "ca-cert": "/path/to/ca-cert.pem",
 "ca-key": "/path/to/ca-key.pem",
 "root-cert": "/path/to/root-cert.pem",
 "cert-chain" "/path/to/cert-chain.pem"
}

curl -d @create_logical_cloud-1-user-2.json http://onap4k8s:<multicloud-k8s_NODE_PORT>/v2/projects/<project-
name>/logical-clouds/control-plane \
        --key ./logical cloud-t1-admin-key.pem \
          --cert ./logical cloud-t1-admin.pem \

Return Status: 201
Return Body:
{
  "name" : "logical-cloud-1"
   "Message" : "logical cloud 1 control plane is successfully created"
}

GET URL: /v2/projects/<project-name>/logical-clouds/<Logical-cloud-name>/control-planes
RETURN STATUS: 200
RETURN BODY:
{
    "name" : "logical-cloud-1",  //name of namespace for the logical cloud
        "gateways" : "istio-egressgateway",
        "dns": "istiocoredns",
        "clusters": {c1, c2}
}

DELETE
URL: /v2/projects/<project-name>/logical-clouds/<Logical-cloud-name>/control-planes

RETURN STATUS: 204
```

Creating new users for the existing Logical cloud

Adding new users in existing Logical cloud 1

user creation

```
URL: /v2/projects/<project-name>/logical-clouds<logical-cloud-name>/users
POST BODY:
{
   "user" : {
    "name" : "user-2",  //name of user for this cloud
    "type" : "certificate",   //type of authentication credentials used by user (certificate, APIKey, UNPW)
    "certificate" : "/path/to/user2/logical cloud-1-user2.csr" ,  //Path to user certificate
    "permissions" : {
       "apiGroups" : ["stable.example.com"]
       "resources" : ["secrets", "pods"]
       "verbs" : ["get", "watch", "list", "create"]
     },
        "quota" : {
                "cpu": "200",
                "memory": "300Gi",
                "pods": "200",
            "dummy/dummyResource": 30,
        }
  }
}

curl -d @create_logical_cloud-1-user-2.json http://onap4k8s:<multicloud-k8s_NODE_PORT>/v2/projects/<project-
name>/logical-clouds \
        --key ./logical cloud-t1-admin-key.pem \
          --cert ./logical cloud-t1-admin.pem \

Return Status: 201
Return Body:
{
  "name" : "logical-cloud-1"
  "user" : "user-2"
  "Message" : "logical cloud and associated user successfully created"
}
```

Tuning Quota for logical cloud

This feature allows the logical cloud to tune their resources.

quota creation

```
URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/quotas
POST BODY:
{
 "cluster-labels": "abc, xyz",
 "resources": {
        "cpu": "400",
        "memory": "1000Gi",
        "pods": "500",
    "dummy/dummyResource": 100,
 }
}

curl -d @create_logical_cloud-1.json http://onap4k8s:<multicloud-k8s_NODE_PORT>/v2/projects/<project-name>
/logical-clouds \
        --key ./logical cloud-t1-admin-key.pem \
          --cert ./logical cloud-t1-admin.pem \

Return Status: 201
Return Body:
{
  "name" : "logical-cloud-1"
   "Message" : "logical cloud 1 is successfully tuned"
}

GET URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/quotas
RETURN STATUS: 200
RETURN BODY:
{
 "resources": {
        "cpu": "400",
        "memory": "1000Gi",
        "pods": "500",
    "dummy/dummyResource": 100,
 }
}

DELETE
URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/quotas

RETURN STATUS: 204
```

Logical cloud Cluster-labels

The following steps explain to get the cluster labels

get cluster labels

```
GET URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/cluster-labels
RETURN STATUS: 200
RETURN BODY:
[{
  "cluster": c1
  "labels" : {abc,xyz,ijk,dfg}
},
{
  "cluster": c2
  "labels" : {abc,xyz,irk,iop}
}
}]
```

Get Logical cloud-config

DCM merge the kube config of each cluster list c1 and c2.

get kubeconfig

```
URL: /v2/projects/<project-name>/logical-clouds/<logical-cloud-name>/kubeconfig
GET
Return Status: 201
Return Body :
{
apiVersion: v1
clusters:
- cluster:
    certificate-authority: path/to/my/cafile
    server: http://2.2.2.2:6443
  name: cluster-abc
- cluster:
    certificate-authority: path/to/my/cafile
    server: https://1.1.1.1:6443
  name: cluster-xyz
contexts:
- context:
    cluster: kubernetes
    namespace: ns-1
    user: user-1
  name: logical-cloud-1
current-context: logical-cloud-1
kind: Config
preferences: {}
users:
- name: user-1
  user:
    client-certificate: path/to/my/client/cert
    client-key: path/to/my/client/key
}
```

Open Questions:

JIRA Story details

Reference

Kubernetes Multi-Tenancy Draft Proposal
Tenant Concept in Kubernetes

Kubernetes Tenant CRD
K8s Multi-tenancy WG Plan