创建、删除环境

2018年12月4日 14:38

命令行添加网络模式sds参数

修改/usr/lib/python2.6/site-packages/fuelclient/objects/environment.py文件Environment类的create方法

```
if net.lower() == "sds":
    data["net_provider"] = "sds"
```

修改/usr/lib/python2.6/site-packages/fuelclient/cli/arguments.py文件的get_net_arg函数

```
choices=("nova", "neutron", "sds")
```

nailgun处理api

POST请求

nailgun/api/v/handlers/cluster.py文件ClusterCollectionHandler类，其父类nailgun/api/v/handlers/base.py文件的CollectionHandler类的POST函数

validator=BasicValidator

1.1.检查数据合法性，调用校验类nailgun/api/v1/validators/base.py文件BasicValidator的checked函数

data=self.checked_data()

校验函数调用类BasicValidator方法validate_json

res=jsonutils.loads(data)解码json数据，将其转化为python对象

1.2调用nailgun/object/cluster.py文件类ClusterCollection的create函数

new_obj=self.collection.create(data)

1.2.1nailgun/object/cluster.py文件ClusterCollection类调用被文件中Cluster类的create函数

```
class ClusterCollection(NailgunCollection):
    single=Cluster
```

1.2.2 objects/cluster.py文件Cluster类的ceate 函数

1) 获取环境变量，获取rel_id、待部署node、fuel_version
2) 调用父类NailgunObject的create函数

new_cluster=super(Cluster,cls).create(data)

①. nailgun/objects/base.py文件NailgunObject类的create函数:在数据库中创建具有指定参数的对象实例

```
new_obj=cls.model()
db().add(new_obj)
db().flush()
```

3) 调用本类方法create_default_group创建nodegroups数据库
4) 调用create_attributes方法为cluster创建属性并生成默认值
5) 调用get_network_manager方法，获取cluster的网络管理器，此处添加sds

```
if instance.net_provider=='sds':
logger.info('debug_nailgun_create_env_08_sds')
from nailgun.network.sds import SdsManager
return SdsManager
```

6) 调用create_network_groups_and_config函数创建网络数据库，配置网络参数，此处添加sds

```
elif cluster.net_provider=='sds':
    cls.create_sds_config(cluster,data.get('net_segment_type'))
```

7) 调用add_pending_changes函数,为cluster添加待定更改

1.1.1 调用BasicValidator类的validate函数

PUT请求.

nailgun/urls.py文件

1) r'/clusters/?$',
ClusterCollectionHandler,#集群集合处理程序
ClusterCollectionHandler定义在api/v1/handlers/cluster.py文件

1.1 类ClusterCollectionHandler定义
collection=objects.ClusterCollection
validator=ClusterValidator

1.1.1 collection = objects.ClusterCollection

调用objects/cluster.py文件的ClusterCollection类
single=Cluster #Cluster是单个集群对象类

1.1.1.1 Cluster类定义objects/cluster.py文件

1）定义数据库模型
model=models.Cluster
调用数据库模型，在db/sqlalchemy/models/cluster.py文件

```
net_provider=Column(
Enum(*consts.CLUSTER_NET_PROVIDERS,name='net_provider'),
nullable=False,
default=consts.CLUSTER_NET_PROVIDERS.nova_network
)
```

此处net_provider的值调用consts.py文件枚举CLUSTER_NET_PROVIDER，此处添加sds如下

```
CLUSTER_NET_PROVIDERS=Enum(
'nova_network',
'neutron',
'sds'
)
```

2） 序列化cluster
serializer=ClusterSerializer
调用objects/serializers/cluster.py文件的ClusterSerializer类（父类BasicSerializer），为fields赋值
父类BasicSerializer

3） 赋值schema

```
"net_provider":{
"type":"string",
"enum":list(consts.CLUSTER_NET_PROVIDERS)
},
```

4）create函数

- 赋值release_id、nodes、fuel_version
- 调用父类objects/base.py文件NailgunObject的create函数，创建数据库
  new_cluster=super(Cluster,cls).create(data)
- 调用create_default_group创建数据库default
- 调用create_attributes

- 根据net_provider的值，导入不同模块，
  cls.get_network_manager(new_cluster).create_network_groups_and_config(new_cluster,data)

```
此处添加sds,调用get_network_manager函数
elif instance.net_provider == 'sds':
    from nailgun.network.sds import SdsManager
    return SdsManager

调用create_network_groups_and_config函数，根据net_provider调用函数，此处添加sds
elif cluster.net_provider == 'sds':
    cls.create_sds_config(cluster,
    data.get('net_segment_type'),
    data.get('net_l23_provider'))
调用SdsManager类的create_sds_config函数进行数据库操作，并读取默认值
sds_config=SdsConfig(
cluster_id=cluster.id,
segmentation_type=segmentation_type,
net_l23_provider=net_l23_provider
)
db().add(sds_config)
meta=cluster.release.networks_metadata["sds"]["config"]
```

通过代码更新数据库

根据版本修改nailgun/db/migration/alembic_migrations/versions下的文件fuel_6_0.py

```
defupgrade():
upgrade_schema()
upgrade_data()
```

a. upgrade函数添加修改

调用upgrade_schema函数更新数据库表和枚举结构

创建sds_config表

```
op.create_table(
    'sds_config',
    sa.Column('id', sa.Integer(), nullable=False),
    sa.Column('vlan_range', JSON(), nullable=True),
    sa.Column('base_mac', LowercaseString(length=17), nullable=False),
    sa.Column('internal_cidr', sa.String(length=48), nullable=True),
    sa.Column('internal_gateway', sa.String(length=48), nullable=True),
    sa.ForeignKeyConstraint(['id'], ['networking_configs.id'], ),
    sa.PrimaryKeyConstraint('id')
)
```

b. 在upgrade_releases函数添加更新的枚举值

更新cluster表单键net_provider枚举的值

```
cluster_net_provider_old = consts.Enum(
    'nova_network',
    'neutron'
)
cluster_net_provider_new = consts.CLUSTER_NET_PROVIDERS
upgrade_enum(
"clusters",
"net_provider",
"net_provider",
cluster_net_provider_old,
cluster_net_provider_new
)
```

其中
CLUSTER_NET_PROVIDERS=Enum(
'nova_network',
'neutron',
'sds'
)

c. downgrade函数调用downgrade_schema添加sds_config
upgrade_enum(
"clusters",
"net_provider",
"net_provider",
cluster_net_provider_old,
cluster_net_provider_new
)