목차

kolla-ansible general configuraion

/etc/kolla/config/global.conf

```
[DEFAULT]
# rpc_response_timeout (def: 60)
rpc_response_timeout: 600
api_limit_max = 1000

# oslo.db config
# https://docs.openstack.org/oslo.db/latest/reference/opts.html
[database]
# Maximum number of SQL connections to keep open in a pool. Setting a value
of 0 indicates no limit. Default) 5
max_pool_size = 10
```

/etc/kolla/config/cinder.conf

```
[DEFAULT]
sf_volume_create_timeout = 600
verify_glance_signatures = disabled
```

/etc/kolla/config/nova.conf

```
[DEFAULT]
block_device_allocate_retries = 1800
block_device_allocate_retries_interval = 3
```

/etc/kolla/config/neutron/ml2_conf.ini

Openvswitch(OVS)환경의 경우

```
[ml2]
type_drivers = flat,vlan,vxlan,geneve
tenant_network_types = vxlan
mechanism_drivers = openvswitch,l2population
extension_drivers = port_security

[ml2_type_vlan]
network_vlan_ranges = physnet1
```

```
[ml2_type_flat]
flat_networks = physnet1

[ml2_type_vxlan]
vni_ranges = 1:1000
```

OVN 환경의 경우

```

[ml2]
type_drivers = flat,vlan,vxlan,geneve
tenant_network_types = geneve
mechanism_drivers = ovn,l2population
extension_drivers = port_security

[ml2_type_vlan]
network_vlan_ranges = physnet1

[ml2_type_flat]
flat_networks = physnet1

[ml2_type_vxlan]
vni_ranges = 1:1000
```

/etc/kolla/config/masakari/masakari-monitors.conf

```

[host]
monitoring_interval = 20

[callback]
retry_max = 2

[introspectiveinstancemonitor]
guest_monitoring_interval = 10
guest_monitoring_timeout = 2
guest_monitoring_failure_threshold = 2
```

배포 스크립트

`deploy.sh`

```

#!/bin/bash

CURR="0"
RELEASE="2024.1"
TARGET="multinode"
```

```
while true; do

    echo "#######################";
    echo -n "0) ping nodes"; if [ $CURR == 0 ]; then echo -n " <== Current";
fi; echo "";
    echo -n "1) bootstrap"; if [ $CURR == 1 ]; then echo -n " <== Current";
fi; echo "";
    echo -n "2) precheck"; if [ $CURR == 2 ]; then echo -n " <== Current";
fi; echo "";
    echo -n "3) deploy"; if [ $CURR == 3 ]; then echo -n " <== Current"; fi;
echo "";
    echo -n "4) post-deploy"; if [ $CURR == 4 ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "5) install client tools"; if [ $CURR == 5 ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "d) destroy"; if [ "$CURR" == "d" ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "p) purge images"; if [ "$CURR" == "p" ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "u) update os"; if [ "$CURR" == "u" ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "r) reboot nodes"; if [ "$CURR" == "r" ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "s) shutdown nodes"; if [ "$CURR" == "s" ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "c) ceph purging"; if [ "$CURR" == "c" ]; then echo -n " <==
Current"; fi; echo "";
    echo -n "m) mariadb recovery"; if [ "$CURR" == "m" ]; then echo -n " <==
Current"; fi; echo "";
    echo "exit) quit";
    echo "#######################";
    echo "Choice) ";

    read x
    if [[ $x = "" ]]; then continue; fi
    CURR=$x;
    case $x in
        exit) break ;;

        0) echo ping nodes...;
    ansible -i ${TARGET} all -m ping;
    ;;

        1) echo Bootstraping...;
        kolla-ansible bootstrap-servers -i ./${TARGET};
        ;;

        2) echo Prechecking... ;
        kolla-ansible prechecks -i ./${TARGET};
        ;;
        3) echo Deploying... ;
        kolla-ansible deploy -i ./${TARGET};
        ;;

        4) echo post-deploy... ;
        kolla-ansible post-deploy -i ./${TARGET};
        ;;

        5) echo install client tools... ;
        pip install python-openstackclient python-cinderclient python-
glanceclient python-novaclient python-neutronclient python-ironicclient
python-designateclient python-heatclient python-manilaclient  python-
swiftclient -c https://releases.openstack.org/constraints/upper/${RELEASE};
        ;;

        d) echo Destroying.. ;
    while true; do
    read -p "Do you wish to DELETE ALL? " yn
    case $yn in
            [Yy]* )
        ansible -m shell -a 'killall qemu-kvm' -i multinode compute;
            kolla-ansible destroy -i ./${TARGET} --yes-i-really-really-mean-
it;
        break;;
            [Nn]* ) break;;
            * ) echo "Please answer yes or no.";;
    esac
    done
        ;;

        p) echo image purging..;
        ansible -m shell -a 'docker rmi $(docker images -q)' -i ./${TARGET}
all;
        ;;

        u) echo update os..;
        ansible -m shell -a 'yum -y update; sync;' -i ./${TARGET} all;
        ;;

    r) echo reboot nodes..;
    ansible -m shell -a 'sync;reboot' -i ./${TARGET} control;
    ansible -m shell -a 'sync;reboot' -i ./${TARGET} compute;
    ;;

    s) echo shutdown nodes..;
    ansible -m shell -a 'sync;shutdown -h now' -i ./${TARGET} control;
    ansible -m shell -a 'sync;shutdown -h now' -i ./${TARGET} compute;
    ;;

    c) echo ceph purging..;
        ansible -m shell -a 'for i in `rados lspools`; do rados purge ${i} -
```

```
-yes-i-really-really-mean-it; done' -i '192.168.41.31,' all
        ;;

        m) echo MariaDB recovery.. ;
        while true; do
        read -p "Do you wish to run MariaDB Recovery? >" yn
        case $yn in
                [Yy]* )
        cd ~;
        echo 'stop control1 mariadb container...';
        ansible -m shell -a 'docker stop mariadb' -i ./${TARGET} control;
        if [ $? -eq 0 ]; then
            echo "mariadb stop successfully"
            kolla-ansible mariadb_recovery -i ./${TARGET};
        else
            echo "mariadb stop failed"
        fi
        break;;

                [Nn]* ) break;;
                * ) echo "Please answer yes or no.";;
        esac
        done
        ;;

        *) echo "Unknown response, enter a number or type 'exit' to quit" ;;
    esac
done
```

DB 복구용 스크립트

mariadb_recovery.sh

```
#!/bin/bash

##############
# recovery 시작하기 전에 control1~3 에서 mariadb container를 종료하여야 한다.
# docker stop mariadb

echo
"###################################################################################"
echo "###### Mariadb Recovery 전에 controller에서 mariadb container를 종료하십시오."
echo "###### mariadb container MUST STOP on all contoller node before run this script."
echo "command: docker stop mariadb"
echo ""
echo "진행하시겠습니까?"
echo "Do you wish to run MariaDB Recovery?"

select yn in "Yes" "No"; do
    case $yn in
        Yes )
    cd ~;
    echo 'stop control1 mariadb container...';
    ansible -m shell -a 'docker stop mariadb' -i multinode control;
    if [ $? -eq 0 ]; then
        echo "mariadb stop successfully"
        kolla-ansible -i multinode mariadb_recovery;
    else
        echo "mariadb stop failed"
    fi
    break;;

        No ) exit;;
    * ) exit;;
    esac
done
```