Introduction

Target of this project is to provide a generic san adapter for eucalyptus (current support 3.3.0/3.3.0.1) . This san adapter comprise of three parties:

- ✧ A block storage manager for storage controller, name "clvm"
- ✧ A customized lvM lock which disable write operation for volume group metadata in node controller host
- ✧ Some patched perl iscsi scripts which change behavior of discovering the exported block device in the host of node controller.

For detail design, please refer to design documents
This adapter only support eucalyptus 3.3.0/3.3.0.1

Compiling

All source codes of this project now can be compiled in centos 6.3 or 6.4, before you begin to compile this project, make sure you already has a environment which can compile eucalyptus 3.3.0/3.3.0.1. for example, all build dependency are resolved

Following are compiling steps:

1) download the source codes from github

#git clone

2) run the build script

#cd gen-san-adapter && ./build.sh

A tarball "gen-san-adapter.tar" will be generated in same directory.

Installation

This generic san adapter need to be installed after you install eucalyptus 3.3.0 and before cloud are initialized

You can have a install package "gen-san-adapter.tar" by compiling the source codes or download from github " https://github.com/nathanxu/gen-san-adapter-tarball"

1) In storage controller

 # tar vxf gen-san-adapter.3.3.0.tar

 # ./install_sc.sh

A jar file will be installed to directory $EUCALYPTUS/usr/share/eucalyptus

2) In node controller

 # tar vxf gen-san-adapter.3.3.0.tar

 # ./install_nc.sh

A lvm lock library will be installed to /lib and perl scripts will be installed into $EUCALYPTUS/usr/share/eucalyptus

Configuration

1) In storage controller

Take a example that you have cluster "cluster001" and you had SAN device attached to SC at /dev/sdb :

 # euca-modify-property -p storage.cluster001.blockmanager=clvm

 # euca-modify-property -p storage.cluster001.sharedevice=/dev/sdb

2) In NC controller

At first, you need to change the lvm configuration

please refer the example of lvm.conf and edit /etc/lvm/lvm.conf by change or adding following items

Configure /etc/lvm/lvm.conf file

In global section. change the locking type and locking library

...

global {

...

 # Type of locking to use. Defaults to local file-based locking (1).

 # Turn locking off by setting to 0 (dangerous: risks metadata corruption

 # if LVM2 commands get run concurrently).

 # Type 2 uses the external shared library locking_library.

 # Type 3 uses built-in clustered locking.

 # Type 4 uses read-only locking which forbids any operations that might

 # change metadata.

#locking_type = 1

locking_type = 2

...

 # The external locking library to load if locking_type is set to 2.

 # locking_library = "liblvm2clusterlock.so"

locking_library = "/lib/liblvm2eucalock.so"

....

}

In activation section, change the volume group filter.

activation {

...

 # If volume_list is defined, each LV is only activated if there is a

 # match against the list.

 # "vgname" and "vgname/lvname" are matched exactly.

 # "@tag" matches any tag set in the LV or VG.

 # "@" matches if any tag defined on the host is also set in the LV or VG*

#

#volume_list = ["vg1", "vg2/lvol1", "@tag1", "@"]*

volume_list = ["@*"]

...

}

In item volume_list, you should add all existing vgs which include non-san pv

for example, in the NC host, if you have volume groups /dev/vg1 and /dev/vg2 which

use the local attached disks

you should configure the volume_list item as:

volume_list = ["vg1","vg2","@*"]

In tags sections, add the following items:

...

tags {

...

hosttags = 1

@192.168.1.101{} #replace "192.168.1.101" as the node controller registry IP

...

}

Configure /etc/iscsi/initiatorname.iscsi file

configure the InitatorName as "InitiatorName=iqn.1994-05.com.redhat:your_node_ctronller_ip

for example, your node controller will be registered in cloud with IP 191.168.1.101, then configure

the InitiatorName as:

InitiatorName=iqn.1994-05.com.redhat:192.168.1.101

Simulate the SAN device

In case that there isn't SAN device in your environment and want to test this adapter. you can use iscsi device to simulate the real SAN device. here is a example.

1. In the host of storage controller. (example: IP 192.168.1.100)

As tgtd already install in host of storage controller. you can make a loop block device then export the loop block device to node controller through iscsi. here is the setups

to make the loop block device and export to node controller

1) make a loop device. (200G),

 # dd if=/dev/zero of=/data/euca_san_block.dat bs=10M count=10240

 # losetup /dev/loop1 /data/euca_san_block.dat

1) export the loop device (target id =1) if not avaible

#tgtadm --lld iscsi --mode account --op new --user test --password test123

 # tgtadm --lld iscsi --op new --mode target --tid 1 -T store1

#tgtadm --lld iscsi --op new --mode logicalunit --tid 1 --lun 1 -b /dev/loop1

#tgtadm --lld iscsi --op bind --mode account --tid 1 --user test

#tgtadm --lld iscsi --op bind --mode target --tid 1 -I ALL

2. In the host of node controller.

In the host of node controller, discovery and connect the ISCSI device.

#iscsiadm -m node -T store1 -p 192.168.1.100 -o update --name node.session.auth.authmethod --value=CHAP

#iscsiadm -m node -T store1 -p 192.168.1.100 --op update --name node.session.auth.username --value=test

#iscsiadm -m node -T store1 -p 192.168.1.100 --op update --name node.session.auth.password --value=test123

#iscsiadm -m node -T store1 -p 192.168.1.100 -l