Technical document on ConPaaS services: ConPaaS MySQL Server

Aleš Černivec

February 27, 2012

Contents

1 Introduction

Currently you can add and remove agent nodes, query for status of the agent nodes, configure users, upload mysql database.

2 Architecture

ConPaaS MySQL Manager node has to be run manually or by the usage of ConPaaS web front-end. ConPaaS Servers can also be managed through direct web front-end (see section 5).

This is only for development: when manager starts, it fetches the fresh package of conpaas sources from public location (e.g. `http://contrail.xlab.si/conpaassql.tar`). Images will be pre-packaged with mysql manager and agent on the release. Template for creating manager, contains details on installation script (`http://contrail.xlab.si/conpaassql/manager/conpaassql-install.sh`, compare to section 4). When installation is complete, manager can be used to orchestrate ConPaaS SQL agents.

When obtaining access point of the manager, new agents can be provisioned by issuing HTTP POST `add_nodes` command on resource `sql-manager-host:/`:

```
POST / HTTP/1.1
Accept: */*
Content-Type: application/json
Content-Length: 67
{"params": {"function": "agent"}, "method": "add_nodes", "id": "1"}
```

Parameter `function` will soon support more than just creating new agent nodes (e.g. cluster manager, cluster agent, cluster). Parameter `method` designates command `add_nodes` and `id` equals 1.

3 Configuration

Configuration of the service consists of two parts:

- configuration file of the service describing **iaas section**, **manager section** and **onevm_agent_template section**
- agent's template file which defines the template used by the manager. The template file is filled in with variables from the configuration file.

An example of the configuration file:

```
[iaas]
DRIVER=OPENNEBULA_XMLRPC
OPENNEBULA_URL=$ONE_URL
OPENNEBULA_USER=$ONE_USERNAME
OPENNEBULA_PASSWORD=$ONE_PASSWORD
OPENNEBULA_IMAGE_ID = $AGENT_IMAGE_ID
OPENNEBULA_SIZE_ID = 1
```

```
OPENNEBULA_NETWORK_ID = $AGENT_NETWORK_ID
OPENNEBULA_NETWORK_GATEWAY=$IP_GATEWAY
OPENNEBULA_NETWORK_NAMESERVER=$NAMESERVER

[manager]
find_existing_agents = true

[onevm_agent_template]
FILENAME=/root/conpaassql/bin/agent.template
NAME=conpaassql_server
CPU=0.2
MEM_SIZE=256
DISK=bus=scsi,readonly=no
OS=arch=i686,boot = hd, root = hda
IMAGE_ID=\$AGENT_IMAGE_ID
NETWORK_ID=$AGENT_NETWORK_ID
CONTEXT=vmid=\$VMID,vmname=\$NAME,ip_private="$NIC[IP, \
  NETWORK=\$NETWORK"]",ip_gateway="\$IP_GATEWAY",netmask=\
  "\$NETMASK",nameserver="\$NAMESERVER",target=sdc,files=\
  \$AGENT_FILES
```

An example of the agent's template file:

```
NAME   = \$NAME
CPU    = \$CPU
MEMORY = \$MEM_SIZE
   OS     = [
   \$OS
   ]
DISK   = [
   IMAGE_ID = \$IMAGE_ID,
   \$DISK
   ]
NIC    = [ NETWORK_ID = \$NETWORK_ID ]
GRAPHICS = [
  type="vnc"
  ]
CONTEXT = [
  \$CONTEXT
  ]
RANK = "- RUNNING_VMS"
```

Agent's template file is referenced within the configuration file. This way the contextualisation of the agent is very generic.

4 Installation on VM images

These steps are necessary in order to clean install ConPaaSSQL server on images used on OpenNebula.

First, you will need

1. `http://contrail.xlab.si/conpaassql/agent/init.sh`
2. `http://contrail.xlab.si/conpaassql/manager/init.sh`

The contents of `init.sh` files is given in appendix 8.

step1

For deploying ConPaaS SQL Manager image, we are using this template description:

```
NAME = conpaas-sql-manager CPU = 0.5 MEMORY = 128
   OS     = [ boot="hd", arch="x86_64" ]

DISK   = [
   source   = "/home/ales/sql/conpaas-sql-1.qcow2",
   target   = "vda",
   readonly = "no",
   DEV_PREFIX = vd,
   DRIVER = qcow2,
   BUS = virtio   # ide (not IDE), SCSI, virtio
]

NIC    = [
   NETWORK = "private-lan"
   ]

GRAPHICS = [
  type="vnc",
  listen="localhost"
  ]

CONTEXT = [
        ip_private  = "\$NIC[IP, NETWORK=\"private-lan\"]",
    ip_gateway  = "10.1.0.254",
    netmask = "255.255.255.0",
    nameserver = "192.168.122.1",
    agent_files="/home/ales/conpaassql/agent/init.sh",
    agent_image_id="205",
    agent_network_id="205",
    one_username=oneadmin,
    one_password=oneadmin,
    one_url="http://10.30.1.14:2633/RPC2"
          files = "/home/ales/conpaassql/manager/init.sh",
          target = "hdb"
]
RANK = "- RUNNING_VMS"
```

It is important to carefully fill in the **context** part of the template. There are several properties to fill in:

- ip_private is manager's IP,
- ip_gateway is manager's gateway,
- netmask is manager's netmask,
- nameserver is manager's domain name server,
- agent_files are files which are used in the agent's **init.sh** script,
- agent_image_id is the agent image's id to be used for provisioning new agents,
- agent_network_id is the agent network id to be used for provisioning new agents,
- one_username is the account used for provisioning new agents,
- one_password is the account used for provisioning new agents,
- one_url is the XMLRPC url to be used by the manager to provision new agents,
- files are the files which appear under the manager as an ISO image,
- target can be left blank (it is for the need for contextualisation).

These context variables are after the booting of the manager used by the **manager's init script** (init.sh, see appendix 8) in order to provision new agent's.

5 ConPaaS MySQL Server Web front-end

ConPaaS MySQL Server Web front-end can easily be installed after the server already runs:

```
# apt-get install python-setuptools python-dev python-pycurl -y
# easy_install pip
# pip install oca apache-libcloud
# pip install --extra-index-url http://eggs.contrail.xlab.si \
conpaassql-manager-gui
# mkdir -p /etc/conpaassql
# cat > /etc/conpaassql/manager-gui.conf << EOF
# MANAGER_HOST = 'localhost'
# EOF
# screen -S conpaasssql-manager-gui -d -m conpaassql_manager_gui
```

It also is not mandatory that the web front-end is installed on the same server as the SQL Server already runs.

6 ConPaaS MySQL Server Manager API

Module `conpaas.mysql.server.manager.internals` contains internals of the ConPaaS MySQL Server. ConPaaS MySQL Server consists of several nodes with different roles.

- Manager node
- Agent node(s)
  - Master
  - Slave(s)

platform: Linux, Debian Squeeze, tested also within Ubuntu 10.10 (there should be no problem when using later distributions).
synopsis: Internals of ConPaaS MySQL Servers.
moduleauthor: Ales Cernivec <ales.cernivec@xlab.si>

`conpaas.mysql.server.manager.internals.add_nodes(kwargs)`

Description:

HTTP POST method. Creates new node and adds it to the list of existing nodes in the manager. A role of new node can be one of: agent, manager. Currently only agent is supported. It makes internal call to `createServiceNodeThread()`.

Parameters:

kwargs – string describing a function (agent).

Returns:

HttpJsonResponse - JSON response with details about the node.

Raises:

ManagerException

Example

```
POST / HTTP/1.1
Accept: */*
Content-Type: application/json

Body content: {"params": {"function": "agent"}, "method": /
"add_nodes", "id": "1"}
```

`conpaas.mysql.server.manager.internals.remove_nodes(params)`

Description:

HTTP POST method. Deletes specific node from a pool of agent nodes. Node deleted is given by `{'serviceNodeId':id}`.

Parameters:

kwargs –string identifying a node.

Returns:

HttpJsonResponse - HttpJsonResponse - JSON response with details about the node. OK if everything went well.

Raises:

ManagerException if something went wrong. It contains a detailed description about the error.

Example

```
POST / HTTP/1.1
Accept: */*
Content-Type: application/json

Body content: {"params": {"serviceNodeId": "12"}, "method": /
"remove_nodes", "id": "1"}
```

`conpaas.mysql.server.manager.internals.list_nodes()`

Description:

HTTP GET method. Uses `IaaSClient.listVMs()` to get list of all service nodes. For each service node it checks if it is in servers list. If some of them are missing they are removed from the list. Returns list of all service nodes.

Parameters:

Returns:

HttpJsonResponse - JSON response with the list of services: `{ 'serviceNode': [<a list of ids>]})`

Raises:

HttpErrorResponse

Example

```
GET /?method=list_nodes&id=1 HTTP/1.1
Accept: */*
Content-Type: application/json
```

`conpaas.mysql.server.manager.internals.get_node_info()`

Description:

HTTP GET method. Gets info of a specific node.

Parameters:

param (str) – serviceNodeId is a VMID of an existing service node.

Returns:

HttpJsonResponse - JSON response with details about the node: : `{'serviceNode':{'id': serviceNode.vmid,'ip': serviceNode.ip,'isRunningMySQL': serviceNode.isRunningMySQL}}`.

Raises:

ManagerException

Example (note the VMID inside the command)

```
GET /?params=%7B%22serviceNodeId%22%3A+%22<VMID>%22%7D&method=get_node_info&id=1 HTTP/1.1
Accept: */*
Content-Type: application/json
```

`conpaas.mysql.server.manager.internals.get_service_info()`

Description:

HTTP GET method. Returns the current state of the manager.

Parameters:

param (str) – serviceNodeId is a VMID of an existing service node.

Returns:

HttpJsonResponse - JSON response with the description of the state.

Raises:

ManagerException

Example

```
GET /?method=get_service_info&id=1 HTTP/1.1
Accept: */*
Content-Type: application/json
```

`conpaas.mysql.server.manager.internals.set_up_replica_master()`

Description:

HTTP POST method. Sets up a replica master node.

Parameters:

id – new replica master id.

Returns:

HttpJsonResponse - JSON response with details about the new node. ManagerException if something went wrong.

Raises:

ManagerException

`conpaas.mysql.server.manager.internals.set_up_replica_slave()`

Description:

HTTP POST method. Sets up a replica master node.

Parameters:

id – new replica slave id.

Returns:

HttpJsonResponse - JSON response with details about the new node. ManagerException if something went wrong.

Raises:

ManagerException

`conpaas.mysql.server.manager.internals.shutdown()`

Description:

HTTP POST method. Shuts down the manager service.

Parameters:

id – new replica slave id.

Returns:

HttpJsonResponse - JSON response with details about the status of a manager node: :py:attr‘S_EPILOGUE‘. ManagerException if something went wrong.

Raises:

ManagerException

`conpaas.mysql.server.manager.internals.get_service_performance()`

Description:

HTTP GET method. Placeholder for obtaining performance metrics.

Parameters:

kwargs (dict) – Additional parameters.

Returns:

HttpJsonResponse – returns metrics

Raises:

Example

```
GET /?method=get_service_performance&id=1 HTTP/1.1
Accept: */*
Content-Type: application/json
```

7 Conclusion

8 Appendix A

In this appendix the content of manager's `init.sh` is given.

```
#!/bin/bash

if [ -f /mnt/context.sh ]; then
  . /mnt/context.sh
fi

ifconfig eth0 \$IP_PRIVATE netmask \$NETMASK
route add default gw \$IP_GATEWAY eth0
echo nameserver \$NAMESERVER > /etc/resolv.conf

# installation
apt-get update
apt-get install -y unzip python python-mysqldb python-pycurl \
 python-dev python-setuptools python-pip subversion

cd /root
svn co svn://svn.forge.objectweb.org/svnroot/contrail/trunk/\
 conpaas/sql/conpaassql
cd conpaassql
python setup.py bdist_egg
easy_install dist/conpaas*

cat > /root/conpaassql/src/conpaas/mysql/server/manager/configuration.cnf << EOF
[iaas]
DRIVER=OPENNEBULA_XMLRPC
OPENNEBULA_URL=$ONE_URL
OPENNEBULA_USER=$ONE_USERNAME
OPENNEBULA_PASSWORD=$ONE_PASSWORD
OPENNEBULA_IMAGE_ID = $AGENT_IMAGE_ID
OPENNEBULA_SIZE_ID = 1
OPENNEBULA_NETWORK_ID = $AGENT_NETWORK_ID
OPENNEBULA_NETWORK_GATEWAY=$IP_GATEWAY
OPENNEBULA_NETWORK_NAMESERVER=$NAMESERVER

[manager]
find_existing_agents = true

[onevm_agent_template]
FILENAME=/root/conpaassql/bin/agent.template
NAME=conpaassql_server
CPU=0.2
MEM_SIZE=256
DISK=bus=scsi,readonly=no
OS=arch=i686,boot = hd, root = hda
IMAGE_ID=$AGENT_IMAGE_ID
NETWORK_ID=$AGENT_NETWORK_ID
```

```
CONTEXT=vmid=$VMID,vmname=$NAME,ip_private="$NIC[IP, NETWORK=$NETWORK"]",\
 ip_gateway="$IP_GATEWAY",netmask="$NETMASK",nameserver="$NAMESERVER",target=sdc,files=$A
EOF

nohup python /root/conpaassql/src/conpaas/mysql/server/manager/server.py -c \
 /root/conpaassql/src/conpaas/mysql/server/manager/configuration.cnf 1>  \
 /var/log/conpaassql-stdout.log 2> /var/log/conpaassql-err.log &
```

The content of agent's `init.sh` is given here:

```
#!/bin/bash

if [ -f /mnt/context.sh ]; then
  . /mnt/context.sh
fi

ifconfig eth0 $IP_PRIVATE netmask $NETMASK
route add default gw $IP_GATEWAY eth0
echo nameserver $NAMESERVER > /etc/resolv.conf

# installation
export DEBIAN_FRONTEND=noninteractive

apt-get update
apt-get install -y unzip python python-mysqldb python-pycurl python-dev \
 python-setuptools python-pip subversion mysql-server supervisor

cd /root
svn co svn://svn.forge.objectweb.org/svnroot/contrail/trunk/conpaas/sql/conpaassql
cd conpaassql
python setup.py bdist_egg
easy_install dist/conpaas*

cat >  /etc/supervisor/conf.d/mysql.conf << EOF
[program:mysqld]
command=/usr/sbin/mysqld
autostart=true
autorestart=true
EOF

cat >>  /etc/supervisor/supervisord.conf << EOF
[inet_http_server]
port = 0.0.0.0:9001
username = root
password = root
EOF

cat >  /root/conpaassql/src/conpaas/mysql/server/agent/configuration.cnf << EOF
[MySQL_root_connection]
```

```
location=
password=contrail
username=root

[MySQL_configuration]
my_cnf_file=/etc/mysql/my.cnf

[supervisor]
user = root
password = root
port = 9001

[ConPaaSSQL]
agent_interface=0.0.0.0
agent_port=60000
manager_ip=0.0.0.0
manager_port=50000
vm_id=10
vm_name=agent10
EOF

# Run the agent command in the background

service supervisor stop
service supervisor start

nohup python /root/conpaassql/src/conpaas/mysql/server/agent/server.py -c \
 /root/conpaassql/src/conpaas/mysql/server/agent/configuration.cnf 1> \
 /var/log/conpaassql-stdout.log 2> /var/log/conpaassql-err.log &
```

9 Appendix B

In this appendix the content of contextualization of the manager and agent nodes' `init.sh` script is given. First, the manager's `init.sh` script:

```
#!/bin/bash

if [ -f /mnt/context.sh ]; then
  . /mnt/context.sh
fi

cat > /home/contrail/sql/src/conpaas/mysql/server/manager/configuration.cnf << EOF
[iaas]
DRIVER=OPENNEBULA_XMLRPC
OPENNEBULA_URL=http://10.30.1.1:2633/RPC2
OPENNEBULA_USER=oneadmin
OPENNEBULA_PASSWORD=oneadmin
OPENNEBULA_IMAGE_ID=/home/contrail/sql/conpaas-sql-1.qcow2
OPENNEBULA_NETWORK_ID=205
OPENNEBULA_SIZE_ID=1
OPENNEBULA_CONTEXT_SCRIPT_MANAGER=/home/contrail/manager/conpaassql-install.sh
OPENNEBULA_CONTEXT_SCRIPT_AGENT=/home/contrail/sql/agent/init.sh
EOF

# Run the manager command in the background
DEST_DIR=/home/contrail/sql
cd ${DEST_DIR}/src
PYTHONPATH=${DEST_DIR}/src:${DEST_DIR}/contrib nohup \
 python conpaas/mysql/server/manager/server.py -c\
 ${DEST_DIR}/src/conpaas/mysql/server/manager/configuration.cnf &
```

The agent's `init.sh` script:

```
#!/bin/bash

if [ -f /mnt/context.sh ]; then
  . /mnt/context.sh
fi

cat >  /home/contrail/sql/src/conpaas/mysql/server/agent/configuration.cnf << EOF
[MySQL_root_connection]
location=
password=contrail
username=root

[MySQL_configuration]
my_cnf_file=/etc/mysql/my.cnf
path_mysql_ssr=/etc/init.d/mysql
EOF

# Run the agent command in the background

DEST_DIR=/home/contrail/sql
cd ${DEST_DIR}/src
PYTHONPATH=${DEST_DIR}/src:${DEST_DIR}/contrib nohup \
 python conpaas/mysql/server/agent/server.py -c\
 ${DEST_DIR}/src/conpaas/mysql/server/agent/configuration.cnf &
```

References