CloudVAMP Documentation

Release 1.0

CloudVAMP

Oct 05, 2017

Contents

1 About CloudVAMP **3**
- 1.1 What is CloudVAMP? . . . 3
- 1.2 Why CloudVAMP? . . . 3
- 1.3 How does CloudVAMP work? . . . 3
- 1.4 What are the technical details? . . . 3

2 Installing CloudVAMP **5**
- 2.1 Getting CloudVAMP from git . . . 5
- 2.2 Installing CVEM . . . 5
- 2.3 Installing the Memory Reporter (MR) . . . 6
- 2.4 Installing the Memory Oversubscription Granter (MOG) . . . 7
- 2.5 Activating CloudVAMP in ONE . . . 8
- 2.6 Activating CloudVAMP for the nodes . . . 8
- 2.7 Tuning CloudVAMP memory management . . . 8

3 Indices and tables **9**

Contents:

CHAPTER 1

About CloudVAMP

What is CloudVAMP?

CloudVAMP stands for "Cloud Virtual Machine Automatic Memory Procurement" and it is an automatic system that enables and manages memory oversubscription in a Cloud on-premises platform based in OpenNebula.

Why CloudVAMP?

Users typically deploy Virtual Machines (VMs) with more resources allocated (e.g. memory) than actually needed by the applications running in the VMs. This might happen for different reasons: i) Users are unaware of the actual requirements of applications due to a lack of profiling; ii) The applications might show dynamic memory consumption during their execution; iii) The Cloud Management Framework (e.g. OpenNebula, OpenStack) determines the amount of memory allocated for the VM out of a set of predefined templates.

How does CloudVAMP work?

CloudVAMP steals the memory that is not used in the running VMs on an OpenNebula Cloud, and makes it available for other VMs. If the memory is later needed by the original VM, it is returned to it.

What are the technical details?

If an excess of free memory for a VM is detected, CloudVAMP changes the amount of memory allocated to the VM via the underlying hypervisor (i.e. KVM) so that it fits the used memory (plus an additional margin of memory), thus considering the running applications. The host that is hosting the VM will then have an extra amount of "stolen memory" that can be used for the deployment of additional VMs on that host. Therefore, CloudVAMP introduces the ability of oversubscription for an on-premises Cloud.

CloudVAMP also prevents overloading, i.e. when the sum of memory allocated to the VMs within a host is greater than the physical amount of memory available for the host. For this, if additional memory is reclaimed by a VM, CloudVAMP uses live migration of VMs across hosts in order to safely prevent overloading without VM downtime.

CHAPTER 2

Installing CloudVAMP

The current version of CloudVAMP is available for OpenNebula. It has been tested in version 4.8 but it is likely to work with newer versions. The installation consists of three steps:

1. Cloud Vertical Elasticity Manager (CVEM), installed in the OpenNebula front-end host.
2. Memory Reporter (MR), installed in the VM
3. Memory Oversubscription Granter (MOG), installed in the OpenNebula front-end host.

Getting CloudVAMP from git

Download CloudVAMP from git:

‘‘‘ $ cd /tmp $ git clone https://github.com/grycap/cloudvamp

‘‘‘

or

‘‘‘ $ cd /tmp $ wget https://github.com/grycap/cloudvamp/archive/master.zip $ unzip cloudvamp-master.zip $ mv cloudvamp-master cloudvamp

‘‘‘

Installing CVEM

Requirements

CVEM uses the library [cpyutils](https://github.com/grycap/cpyutils). Follow the instructions there to install it.

Install in the system path

Enter the CVEM directory in the downloaded files and perform the setup installation.

‘‘‘ $ cd /tmp/cloudvamp/cvem $ python setup install

```
```

Install in a specific path

Select a proper path where to install the CVEM service (i.e. */usr/local/cvem*, */opt/cvem* or other). This path will be called *CVEM_*

```
$ cp -r cd /tmp/cloudvamp/cvem /usr/local
```

Finally you must copy (or link) *$CVEM_PATH/scripts/cvem* file to */etc/init.d* directory.

```
$ ln -s /usr/local/cvem/scripts/cvem /etc/init.d
```

Configuration

If you want the CVEM service to be automatically started at boot time, execute the following commands:

On RedHat Systems:

```
$ chkconfig cvem on
```

On Debian Systems:

```
$ update-rc.d cvem start 99 2 3 4 5 . stop 05 0 1 6 .
```

Or you can do it manually:

```
$ ln -s /etc/init.d/cvem /etc/rc2.d/S99cvem $ ln -s /etc/init.d/cvem /etc/rc3.d/S99cvem $ ln -s /etc/init.d/cvem
/etc/rc5.d/S99cvem $ ln -s /etc/init.d/cvem /etc/rc1.d/K05cvem $ ln -s /etc/init.d/cvem /etc/rc6.d/K05cvem
```

Adjust the installation path by setting the DAEMON variable at */etc/init.d/cvem* to the path where the CVEM *cvemd.py* file is installed (e.g. */usr/local/cvem/cvemd.py*), or set the name of the script file (*cvemd.py*) if the file is in the environment variable *PATH*.

Finally edit the configurations files *.cfg located in the etc/ directory where the CVEM is installed (e.g */opt/cvem/etc*) or in the */etc/cvem* directory:

- *cvem.cfg*: CVEM configuration file.
- *one.cfg*: OpenNebula specific configuration file.

Installing the Memory Reporter (MR)

The memory reporter relies on the OpenNebula contextualization scripts. Therefore, it must be installed on the Virtual Machine images used in the platform. See more information in the [OpenNebula Documentation](http://docs.opennebula.org/4.12/user/virtual_machine_setup/bcont.html).

It uses the OneGate system to publish and get the monitored information so OneGate must be installed and configured, as indicated in the [OpenNebula's OneGate Documentation](http://archives.opennebula.org/documentation:rel4.4:onegate_usage).

The first step is to upload the contextualization scripts located in */tmp/cloudvamp/onegate_scripts/* (*onegate_init.sh*, *onegate_publisher.sh*) to the OpenNebula "Files & Kernels" section.

Then all the templates of the VMs must be configured to activate the OpenNebula contextualization Token, include both files, and set *onegate_init.sh* as an init script:

“‘

CONTEXT = [FILES_DS = “$FILE[IMAGE_ID=<init_file_id>] $FILE[IMAGE_ID=<publisher_file_id>]”, INIT_SCRIPTS = “onegate_init.sh”, TOKEN = “YES”]

“‘

Installing the Memory Oversubscription Granter (MOG)

Creating the IM and the VMM

The MOG, installed in the OpenNebula front-end node, consists of two parts: an Information Manager (IM) and a Virtual Machine Manager (VMM), which are concepts from OpenNebula. The code has been tested in Ubuntu, with a ONE 4.8 installation from the official OpenNebula repositories.

Create a folder for the CloudVAMP VMM at ONE’s vmm folder (usually */var/lib/one/remotes/vmm*) and create links to the files in the kvm folder:

“‘ $ mkdir /var/lib/one/remotes/vmm/cloudvamp $ cd /var/lib/one/remotes/vmm/cloudvamp

“‘

Now copy the new files in the folder

“‘ $ cp /tmp/cloudvamp/opennebula/vmm/cloudvamp/* /var/lib/one/remotes/vmm/cloudvamp/

“‘

Finally link the other file in the folder (please ignore the messages that state that the files already exist):

“‘ $ ln -s ../kvm/* /var/lib/one/remotes/vmm/cloudvamp/

“‘

Create a folder for cloudvamp im probes files at ONE’s im folder (usually */var/lib/one/remotes/im*) and create links to the files in kvm folder:

“‘ $ mkdir /var/lib/one/remotes/im/cloudvamp-probes.d

“‘

Now copy the new files in the folder

“‘ $ cp /tmp/cloudvamp/opennebula/im/cloudvamp-probes.d/* /var/lib/one/remotes/im/cloudvamp-probes.d/

“‘

Link the files to the kvm probes and set the values for the cloudvamp probes

“‘ $ ln -s /var/lib/one/remotes/im/kvm-probes.d/* /var/lib/one/remotes/im/cloudvamp-probes.d/ $ cd /var/lib/one/remotes/im/cloudvamp-probes.d $ rm kvm.rb $ chmod 644 poll.sh

“‘

- NOTE: take into account that the file “poll.sh” must not have permission for execution (i.e. its permission mask is 0644).

Finally, create a folder for the cloudvamp im as a link to the kvm one:

“‘ $ ln -s /var/lib/one/remotes/im/kvm.d /var/lib/one/remotes/im/cloudvamp.d

“‘

Activating CloudVAMP in ONE

Edit the one configuration file (i.e. */etc/one/oned.conf*) and add the following lines

“‘

IM_MAD = [name = “cloudvamp”, executable = “one_im_ssh”, arguments = “-r 3 -t 15 cloudvamp”]

VM_MAD = [name = “cloudvamp”, executable = “one_vmm_exec”, arguments = “-t 15 -r 0 cloudvamp”, default = “vmm_exec/vmm_exec_kvm.conf”, type = “kvm”]

“‘

And restart ONE ` $ su - oneadmin $ one restart `

Activating CloudVAMP for the nodes

Now you should delete (or deactivate) the nodes under the control of ONE and create them again using the cloudvamp im and vmm. This will only apply to these VMs that are running the KVM hypervisor

> In this particular case we are using dummy network

```
` $ onehost disable myhost01 $ onehost create myhost01 -i cloudvamp
-v cloudvamp -n dummy `
```

Tuning CloudVAMP memory management

In case that you do not want that all the memory stolen from the running VMs is dedicated to other VMs, you can tune the $O value from file */var/lib/one/remotes/im/cloudvamp-probes.d/cloudvamp.rb*.

> E.g. If you want that only the 80% of borrowed memory is dedicated for possible oversubscription, please search for the line $O = 1.0 and set it to $O = 0.8 (take into account that it is a floating point number, in case that you get back to 100%)

CHAPTER 3

Indices and tables

- genindex
- search