ceilometer-powervm Documentation

Release 7.0.0.0rc2.dev2

IBM

Aug 14, 2018

Contents

This project will provide Ceilometer-compatible compute agent plugins for monitoring instance utilization and statistics on PowerVM systems.

Documentation on Ceilometer can be found at the Ceilometer documentation.

CHAPTER 1

Ceilometer-PowerVM Information and Configuration

Contents:

1.1 Team and repository tags

1.2 Support for PowerVM Performance Monitoring

The IBM PowerVM hypervisor provides virtualization on POWER hardware. PowerVM customers can see benefits in their environments by making use of OpenStack. This project implements a Ceilometer-compatible compute inspector. This inspector, along with the PowerVM Nova driver and Neutron agent, provides capability for PowerVM customers to natively monitor utilization and statistics for instances running on OpenStack-managed systems.

1.2.1 Problem Description

PowerVM supports a variety of performance monitoring interfaces within the platform, providing virtual machine and system monitoring data. Ceilometer-powervm implements a Ceilometer-based compute inspector for the PowerVM hypervisor.

1.2.2 Inspector Description

The Ceilometer compute agent provides an inspector framework that allows hypervisors to integrate support for gathering instance statistics and utilization details into Ceilometer. This project provides a standard Ceilometer virt inspector that pulls its data from the PowerVM Performance and Capacity Monitoring (PCM) infrastructure.

This inspector retrieves instance monitoring data for cpu, network, memory, and disk usage. Interactions with PowerVM PCM occur using the PowerVM REST API stack through pypowervm, an open source python project.

This inspector requires that the PowerVM system be configured for management via NovaLink.

End User Impact

The users of the cloud are able to see the metrics for their virtual machines. As PowerVM deals with 'disk buses' rather than specific disks, the hard disk data is reported at a 'per bus' level (i.e. each SCSI or Virtual Fibre Channel bus).

Performance/Scalability Impacts

None.

Other deployer impact

The cloud administrator needs to install the ceilometer-powervm project on their PowerVM compute node. It must be installed on the NovaLink virtual machine on the PowerVM system.

The cloud administrator needs to configure their 'hypervisor_inspector' as powervm.

No other configuration is required.

Developer impact

None

1.2.3 Implementation

Assignee(s)

Primary assignee: thorst

Ongoing maintainer: thorst

1.2.4 Future lifecycle

Ongoing maintenance of the PowerVM compute inspector will be handled by the IBM OpenStack team.

1.2.5 Dependencies

- The Ceilometer compute agent.
- The pypowervm library.
- A NovaLink enabled PowerVM system.

1.2.6 References

- Ceilometer Architecture: http://docs.openstack.org/developer/ceilometer/architecture.html
- pypowervm: https://github.com/powervm/pypowervm
- NovaLink: http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=AN&subtype=CA&htmlfid=897/ENUS215-262&appname=USN
- PowerVM REST API Initial Specification (may require a newer version as they become available): http://ibm.co/1lThV9R
- PowerVM Virtualization Introduction and Configuration: http://www.redbooks.ibm.com/abstracts/sg247940.html?Open
- PowerVM Best Practices: http://www.redbooks.ibm.com/abstracts/sg248062.html?Open

CHAPTER 2

Ceilometer-PowerVM Policies

Contents:

2.1 Ceilometer-PowerVM Policies

In the Policies Guide, you will find documented policies for developing with Ceilometer-PowerVM. This includes the processes we use for blueprints and specs, bugs, contributor onboarding, and other procedural items.

2.1.1 Policies

Ceilometer-PowerVM Bugs

Ceilometer-PowerVM maintains all of its bugs in Launchpad. All of the current open Ceilometer-PowerVM bugs can be found in that link.

Bug Triage Process

The process of bug triaging consists of the following steps:

1. Check if a bug was filed for a correct component (project). If not, either change the project or mark it as "Invalid".
2. Add appropriate tags. Even if the bug is not valid or is a duplicate of another one, it still may help bug submitters and corresponding sub-teams.
3. Check if a similar bug was filed before. If so, mark it as a duplicate of the previous bug.
4. Check if the bug description is consistent, e.g. it has enough information for developers to reproduce it. If it's not consistent, ask submitter to provide more info and mark a bug as "Incomplete".
5. Depending on ease of reproduction (or if the issue can be spotted in the code), mark it as "Confirmed".

6. Assign the importance. Bugs that obviously break core and widely used functionality should get assigned as “High” or “Critical” importance. The same applies to bugs that were filed for gate failures.
7. (Optional). Add comments explaining the issue and possible strategy of fixing/working around the bug.

Contributing to Ceilometer-PowerVM

If you would like to contribute to the development of OpenStack, you must follow the steps in the “If you’re a developer, start here” section of this page:

> http://wiki.openstack.org/HowToContribute

Once those steps have been completed, changes to OpenStack should be submitted for review via the Gerrit tool, following the workflow documented at:

> http://wiki.openstack.org/GerritWorkflow

Pull requests submitted through GitHub will be ignored.

Bugs should be filed on Launchpad, not GitHub:

> https://bugs.launchpad.net/ceilometer-powervm

Code Reviews

Code reviews are a critical component of all OpenStack projects. Code reviews provide a way to enforce a level of consistency across the project, and also allow for the careful onboarding of contributions from new contributors.

Code Review Practices

Ceilometer-PowerVM follows the code review guidelines as set forth for all OpenStack projects. It is expected that all reviewers are following the guidelines set forth on that page.

Team and repository tags

Support for PowerVM Performance Monitoring

The IBM PowerVM hypervisor provides virtualization on POWER hardware. PowerVM customers can see benefits in their environments by making use of OpenStack. This project implements a Ceilometer-compatible compute inspector. This inspector, along with the PowerVM Nova driver and Neutron agent, provides capability for PowerVM customers to natively monitor utilization and statistics for instances running on OpenStack-managed systems.

Problem Description

PowerVM supports a variety of performance monitoring interfaces within the platform, providing virtual machine and system monitoring data. Ceilometer-powervm implements a Ceilometer-based compute inspector for the PowerVM hypervisor.

Inspector Description

The Ceilometer compute agent provides an inspector framework that allows hypervisors to integrate support for gathering instance statistics and utilization details into Ceilometer. This project provides a standard Ceilometer virt inspector that pulls its data from the PowerVM Performance and Capacity Monitoring (PCM) infrastructure.

This inspector retrieves instance monitoring data for cpu, network, memory, and disk usage. Interactions with PowerVM PCM occur using the PowerVM REST API stack through pypowervm, an open source python project.

This inspector requires that the PowerVM system be configured for management via NovaLink.

End User Impact

The users of the cloud are able to see the metrics for their virtual machines. As PowerVM deals with 'disk buses' rather than specific disks, the hard disk data is reported at a 'per bus' level (i.e. each SCSI or Virtual Fibre Channel bus).

Performance/Scalability Impacts

None.

Other deployer impact

The cloud administrator needs to install the ceilometer-powervm project on their PowerVM compute node. It must be installed on the NovaLink virtual machine on the PowerVM system.

The cloud administrator needs to configure their 'hypervisor_inspector' as powervm.

No other configuration is required.

Developer impact

None

Implementation

Assignee(s)

Primary assignee: thorst

Ongoing maintainer: thorst

Future lifecycle

Ongoing maintenance of the PowerVM compute inspector will be handled by the IBM OpenStack team.

Dependencies

- The Ceilometer compute agent.
- The pypowervm library.
- A NovaLink enabled PowerVM system.

References

- Ceilometer Architecture: http://docs.openstack.org/developer/ceilometer/architecture.html
- pypowervm: https://github.com/powervm/pypowervm
- NovaLink: http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?infotype=AN&subtype=CA&htmlfid=897/ENUS215-262&appname=USN
- PowerVM REST API Initial Specification (may require a newer version as they become available): http://ibm.co/1lThV9R
- PowerVM Virtualization Introduction and Configuration: http://www.redbooks.ibm.com/abstracts/sg247940.html?Open
- PowerVM Best Practices: http://www.redbooks.ibm.com/abstracts/sg248062.html?Open

2.1.2 Indices and tables

- genindex
- modindex
- search

CHAPTER 3

Ceilometer-PowerVM Devref

Contents:

3.1 Developer Guide

In the Developer Guide, you will find information on how to develop for Ceilometer-PowerVM and how it interacts with Ceilometer. You will also find information on setup and usage of Ceilometer-PowerVM.

3.1.1 Internals and Programming

Setting Up a Development Environment

This page describes how to setup a working Python development environment that can be used in developing ceilometer-powervm.

These instructions assume you're already familiar with Git and Gerrit, which is a code repository mirror and code review toolset, however if you aren't please see this Git tutorial for an introduction to using Git and this guide for a tutorial on using Gerrit and Git for code contribution to Openstack projects.

Getting the code

Grab the code:

```
git clone git://git.openstack.org/stackforge/ceilometer-powervm
cd ceilometer-powervm
```

Setting up your environment

The purpose of this project is to provide the 'glue' between OpenStack Telemetry (Ceilometer) and PowerVM. The pypowervm project is used to control and monitor PowerVM systems.

It is recommended that you clone down the OpenStack Ceilometer project along with pypowervm into your respective development environment.

Running the tox python targets for tests will automatically clone these down via the requirements. When run with tox, it pulls the necessary requirements into a virtualenv.

Additional project requirements may be found in the requirements.txt file.

Usage

- Configure the PowerVM system for NovaLink
- Install the ceilometer-powervm plugin on the NovaLink VM on the PowerVM Server.
- Set the hypervisor_inspector in the ceilometer.conf to "powervm"
- Start the ceilometer-agent-compute on the compute server

3.1.2 Testing

Running Ceilometer-PowerVM Tests

This page describes how to run the Ceilometer-PowerVM tests. This page assumes you have already set up an working Python environment for Ceilometer-PowerVM development.

With *tox*

Ceilometer-PowerVM, like other OpenStack projects, uses tox for managing the virtual environments for running test cases. It uses Testr for managing the running of the test cases.

Tox handles the creation of a series of virtualenvs that target specific versions of Python.

Testr handles the parallel execution of series of test cases as well as the tracking of long-running tests and other things.

For more information on the standard tox-based test infrastructure used by OpenStack and how to do some common test/debugging procedures with Testr, see this wiki page:

> https://wiki.openstack.org/wiki/Testr

PEP8 and Unit Tests

Running pep8 and unit tests is as easy as executing this in the root directory of the Ceilometer-PowerVM source code:

```
tox
```

To run only pep8:

```
tox -e pep8
```

Since pep8 includes running pylint on all files, it can take quite some time to run. To restrict the pylint check to only the files altered by the latest patch changes:

```
tox -e pep8 HEAD~1
```

To run only the unit tests:

```
tox -e py27
```

3.1.3 Indices and tables

- genindex
- modindex
- search