Pushing Prime Performance Manager Data to Other Applications

Although Prime Performance Manager is designed to get data from devices and display that data in a variety of ways, you can push Prime Performance Manager data to certain applications including Graphite and OpenStack. Procedures for pushing data to other applications are provided in the following topics:

Pushing Prime Performance Manager Reports to Graphite

Graphite is an open-source tool that monitors and graphs computer performance. It stores numeric time-series data and displays data graphs on demand. Graphite has three components:

- Carbon—A twisted daemon that listens for time-series data.
- Whisper—A simple database library for storing time-series data similar in design to the round-robin database tool (RRDTool).
- Graphite webapp—A Django web application that renders graphs on-demand using Cairo, a 2D vector graphics API used by Firefox and other desktop projects.

To display Prime Performance Manager reports in Graphite you must push the report data to Carbon:

Step 1 Log into the Prime Performance Manager as the root user. See Logging In as the Root User, page 2-1.

Step 2 Verify that devices are imported into Prime Performance Manager and reports are being generated successfully.

Step 3 Navigate to the /opt/CSCOppm-gw/etc/nbapps directory.

Step 4 Open the pushToCarbon.xml file with a text editor.

Step 5 Modify the following pushToCarbon.xml parameters:

- Listen—The Prime Performance Manager database table that stores the report data.
- Interval—The report interval.
- Enabled—Enables or disables pushing of report data to Carbon. Set to **true** for Prime Performance Manager to begin pushing data.
- PushToCarbon(carbonIP, carbonPort, tableName, KPIName, [queueSize], [reconnectInterval])

You can customize the Carbon database table name to include the following information in the name:

- [DeviceName]—Name of the device.
- [BaseTableName]—Name of the Prime Performance Manager database table.
- [KeyNames]—Name of the Prime Performance Manager database table key.
- [Interval]—Report interval.

Note Dots (".") are not allowed in pushToCarbon.xml fields. If you enter dots, they will be replaced with underscores ("_"). For example, if a device name is 172.18.116.16, in Carbon it will be changed to 172_18_116_16.

Example

A configured pushTo Carbon.xml example is shown below:

```
<?xml version="1.0"?>
<!-- Copyright (c) 2012 Cisco Systems, Inc. All rights reserved. -->
<ns:PushList
    xmlns:ns="http://cisco.com/ppm/poller">
    <Push name="carbonKVMDomainAvailability" listen="kvmDomainAvailability"
location="gateway" interval="Min1" enabled="false">
        <Collection>
            PushToCarbon("172.18.116.18", "2003",
"local.[DeviceName].[BaseTableName].[KeyNames].[Interval]", "Availability");
        </Collection>
    </Push>

    <Push name="carbonKVMDomainTotalCPU" listen="kvmDomainTotalCPU" location="gateway"
interval="Min1" enabled="false">
        <Processing>
            CPUTime = CPUTime / IntervalDuration() / 1000000000 / VCPUs;
        </Processing>
        <Collection>
            PushToCarbon("172.18.116.18", "2003",
"local.[DeviceName].[BaseTableName].[KeyNames].[Interval]", "CPUTime");
        </Collection>
    </Push>
</ns:PushList>
```

Pushing Prime Performance Manager Data to OpenStack

Prime Performance Manager can monitor OpenStack virtual machines (VMs) through different paths and therefore retrieve some VM data not available in OpenStack Ceilometers. For example, Prime Performance Manager can get the availability percentage of each OpenStack VM. By pushing this data to the OpenStack Ceilometers, OpenStack users will see the new meter created and supported by Prime Performance Manager in the Ceilometer.

To push data to OpenStack Ceilometers, you must modify pushToCeilometer.xml for each push processor. pushToCeilometer.xml is located in the /opt/CSCOppm-gw/etc/nbapps/ directory**.**

Open pushToCeilometer.xml with a text editor and enter the following:

- listen—Enter the data source.

- interval—Enter the interval of report data to be pushed. The interval must be enabled for the data source and be a valid interval value defined in EventPoller.xsd. By default it includes all user-configurable intervals.
- queueSize—Defines the queue size. The default is 10000.
- reconnectInterval—If Prime Performance Manager loses connection to the Ceilometer while it is pushing data, it will put the message in a queue and keep trying to reconnect. reconnectInterval defines the interval between reconnect attempts. The default value is 60000 ms, which means Prime Performance Manager will try to reconnect every 60 seconds until the connection is reestablished.
- Usage of macro—PushToCeilometer(tableName, KPIName, Unit, [queueSize], [reconnectInterval])

In the following expressions, default values for queueSize and reconnectInterval are used:

- PushToCeilometer("ppmKVMDomainCPU", "CPUTime", "%");
- PushToCeilometer("ppmKVMDomainCPU", "CPUTime", "%", "default", "default");

In the following expressions, both custom and default values for queueSize and reconnectInterval are used:

- PushToCeilometer("ppmKVMDomainCPU", "CPUTime", "%", "5000", "30000");
- PushToCeilometer("ppmKVMDomainCPU", "CPUTime", "%", "5000", "default");
- PushToCeilometer("ppmKVMDomainCPU", "CPUTime", "%", "default", "30000");

All pushing processors share one queue and one connection. If you define different queueSize and reconnectIntervals in the expressions, the largest queueSize and the smallest reconnectInterval are used. For example, in the following expression set, queueSize=5000 and reconnectInterval=30000 will be used:

```
    <Push name="kvmDomainAvailability" listen="kvmDomainAvailability" location="gateway"
interval="Min1" enabled="true">
        <Collection>
            PushToCeilometer("ppmKVMDomainAvailability", "Availability", "%", "3000",
"30000");
        </Collection>
    </Push>

    <Push name="kvmDomainTotalCPU" listen="kvmDomainTotalCPU" location="gateway"
interval="Min1" enabled="true">
        <Processing>
            CPUTime = CPUTime / IntervalDuration() / 1000000000 / VCPUs;
        </Processing>
        <Collection>
            PushToCeilometer("ppmKVMDomainCPU", "CPUTime", "%", "5000", "50000");
        </Collection>
</Push>
```

After modifying and saving the pushToCeilometer.xml, Prime Performance Manager pushes the defined VM reports to the integrated OpenStack Ceilometer.