# Red Hat Satellite 6.15

## Monitoring Satellite performance

Collect metrics from Satellite and allow their analysis in external tools

Last Updated: 2024-06-19

# Red Hat Satellite 6.15 Monitoring Satellite performance

Collect metrics from Satellite and allow their analysis in external tools

Red Hat Satellite Documentation Team
satellite-doc-list@redhat.com

## Legal Notice



## Abstract

This guide describes how to gather metrics from Red Hat Satellite 6 for analysis. It is aimed at Satellite administrators.

# Table of Contents

# MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. Because of the enormity of this endeavor, these changes are being updated gradually and where possible. For more details, see our CTO Chris Wright's message .

# PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

Use the **Create Issue** form in Red Hat Jira to provide your feedback. The Jira issue is created in the Red Hat Satellite Jira project, where you can track its progress.

**Prerequisites**

- Ensure you have registered a Red Hat account.

**Procedure**

1. Click the following link: **Create Issue**. If Jira displays a login error, log in and proceed after you are redirected to the form.
2. Complete the **Summary** and **Description** fields. In the **Description** field, include the documentation URL, chapter or section number, and a detailed description of the issue. Do not modify any other fields in the form.
3. Click **Create**.

# CHAPTER 1. METRICS OVERVIEW

You can set up a third-party solution to collect live metrics from Satellite Server, archive them for a fixed period of time, and analyze them. Obtaining metrics from Satellite is useful for capacity planning and troubleshooting performance issues.

If you need to raise a support case with Red Hat to resolve a performance issue, the archived data provides valuable insight. Note that Red Hat Support can only access the archived data if you upload it to a support case.

You can collect the following metrics from Satellite:

- Basic statistics from the operating system, including system load, memory utilization, and input/output operations
- Process statistics, including memory and CPU utilization
- Apache HTTP Server activity statistics
- PostgreSQL activity statistics
- Satellite application statistics

# CHAPTER 2. METRICS SOLUTION COMPONENTS

Red Hat recommends using the Performance Co-Pilot to collect and archive Satellite metrics.

**Performance Co-Pilot (PCP)**

Performance Co-Pilot is a suite of tools and libraries for acquiring, storing, and analyzing system-level performance measurements. You can use PCP to analyze live and historical metrics in the CLI.

**Performance Metric Domain Agents (PMDA)**

A Performance Metric Domain Agent is a PCP add-on which enables access to metrics of an application or service. To gather all metrics relevant to Satellite, you have to install PMDA for Apache HTTP Server and PostgreSQL.

**Grafana**

A web application that visualizes metrics collected by PCP. To analyze metrics in the web UI, you have to install Grafana and the Grafana PCP plugin.

# CHAPTER 3. SETTING UP THE METRICS MONITORING SOLUTION

Install PCP packages and configure PCP data collection. You can use the PCP CLI tools to retrieve metrics in the command line. Optionally, you can install Grafana to enable web UI access to metrics.

## 3.1. INSTALLING PCP

Install the PCP packages on your Satellite Server and enable PCP daemons.

### Prerequisites

- Ensure you have the minimum of 20 GB space available in the **/var/log/pcp** directory.
  With the default PCP data retention settings, data storage is estimated to use between 100 MB and 500 MB of disk space per day, but may use up to several gigabytes over time. For more information, see Chapter 4, *Metrics data retention*.

### Procedure

1. Install the PCP packages:

```
# satellite-maintain packages install pcp \
pcp-pmda-apache \
pcp-pmda-openmetrics \
pcp-pmda-postgresql \
pcp-pmda-redis \
pcp-system-tools \
foreman-pcp
```

2. Enable and start the Performance Metrics Collector daemon and Performance Metrics Logger daemon:

```
# systemctl enable --now pmcd pmlogger
```

## 3.2. CONFIGURING PCP DATA COLLECTION

You can configure PCP to collect metrics about processes, Satellite, Apache HTTP Server, and PostgreSQL.

### Procedure

1. Symlink the Satellite specific configuration to PMDA process monitoring:

```
# ln -s /etc/pcp/proc/foreman-hotproc.conf /var/lib/pcp/pmdas/proc/hotproc.conf
```

   By default, PCP only collects basic system metrics. This step enables detailed metrics about the following Satellite processes:

   - Java
   - PostgreSQL

- Redis
- Dynflow
- Puma
- Pulpcore

2. Install the process monitoring PMDA:

```
# cd /var/lib/pcp/pmdas/proc
# ./Install
```

3. Configure PCP to collect metrics from Apache HTTP Server.
   a. Enable the Apache HTTP Server extended status module:

```
# satellite-installer --enable-apache-mod-status
```

   b. Enable the Apache HTTP Server PMDA:

```
# cd /var/lib/pcp/pmdas/apache
# ./Install
```

4. Configure PCP to collect metrics from PostgreSQL:

```
# cd /var/lib/pcp/pmdas/postgresql
# ./Install
```

5. Enable the telemetry feature in Satellite:

```
# satellite-installer --foreman-telemetry-prometheus-enabled true
```

6. Configure PCP to collect data from Satellite:

```
# cd /var/lib/pcp/pmdas/openmetrics
# echo "https://satellite.example.com/metrics" > config.d/foreman.url
# ./Install
```

7. Restart PCP to begin data collection:

```
# systemctl restart pmcd pmlogger
```

## 3.3. VERIFYING PCP CONFIGURATION

You can verify that PCP is configured correctly and services are active.

**Procedure**

- Print a summary of the active PCP configuration:

```
# pcp
```

Example output of the **pcp** command:

```
Performance Co-Pilot configuration on satellite.example.com:

 platform: Linux satellite.example.com 4.18.0-372.32.1.el8_6.x86_64 #1 SMP Fri Oct 7
12:35:10 EDT 2022 x86_64
 hardware: 16 cpus, 2 disks, 1 node, 31895MB RAM
 timezone: UTC
 services: pmcd pmproxy
    pmcd: Version 5.3.7-17, 13 agents, 4 clients
    pmda: root pmcd proc pmproxy xfs redis linux apache mmv kvm
          postgresql jbd2 openmetrics
 pmlogger: primary logger: /var/log/pcp/pmlogger/satellite.example.com/20230831.00.10
    pmie: primary engine: /var/log/pcp/pmie/satellite.example.com/pmie.log
```

In this example, both the Performance Metrics Collector Daemon (pmcd) and Performance Metrics Proxy Daemon (pmproxy) services are running. It also confirms the PMDA that are collecting metrics. Finally, it lists the active log file, in which **pmlogger** is currently storing metrics.

## 3.4. ENABLING WEB UI ACCESS TO METRICS

You can enable web UI access to metrics collected by PCP by installing Grafana.

**Procedure**

1. Install Grafana and the Grafana PCP plugin on your Satellite Server:

```
# satellite-maintain packages install grafana grafana-pcp
```

2. Start and enable the Grafana web service and the PCP proxy service:

```
# systemctl enable --now pmproxy grafana-server
```

3. Open the firewall port to allow access to the Grafana web interface:

```
# firewall-cmd --permanent --add-service=grafana
```

4. Reload the firewall configuration to apply the changes:

```
# firewall-cmd --reload
```

5. Install PCP Redis and configure Grafana to load it. For more information, see Configuring PCP Redis in *Red Hat Enterprise Linux 8 Monitoring and managing system status and performance* .

6. Access the Grafana web UI, enable the PCP plugin, and add PCP Redis as a data source. For more information, see Accessing the Grafana web UI in *Red Hat Enterprise Linux 8 Monitoring and managing system status and performance*.

# CHAPTER 4. METRICS DATA RETENTION

The storage capacity required by PCP data logging is determined by the following factors:

- The logged metrics
- The logging interval
- The retention policy

The default logging (sampling) interval is 60 seconds. The default retention policy is to compress archives older than one day and to keep archives for the last 14 days.

You can increase the logging interval or shorten the retention policy to save storage space. If you require high-resolution sampling, you can decrease the logging interval. In such case, ensure that you have enough storage space.

PCP archive logs are stored in the **/var/log/pcp/pmlogger/*satellite.example.com*** directory.

## 4.1. CHANGING DEFAULT LOGGING INTERVAL

You can change the default logging interval to either increase or decrease the sampling rate, at which the PCP metrics are logged. A larger interval results in a lower sampling rate.

**Procedure**

1. Open the **/etc/pcp/pmlogger/control.d/local** configuration file.
2. Locate the **LOCALHOSTNAME** line.
3. Append **-t *XX*s**, where *XX* is the required time interval in seconds.
4. Save the file.
5. Restart the **pmlogger** service:

```
# systemctl restart pmlogger
```

## 4.2. CHANGING DATA RETENTION POLICY

You can change the data retention policy to control after how long the PCP data are archived and deleted.

**Procedure**

1. Open the **/etc/sysconfig/pmlogger_timers** file.
2. Locate the **PMLOGGER_DAILY_PARAMS** line.
3. If the line is commented, uncomment the line.
4. Configure the following parameters:
   - Ensure the default **-E** parameter is present.

- Append the **-x** parameter and add a value for the required number of days after which data is archived.
- Append the **-k** parameter and add a value for the number of days after which data is deleted.

For example, the parameters **-x 4 -k 7** specify that data will be compressed after 4 days and deleted after 7 days.

5. Save the file.

## 4.3. VIEWING DATA STORAGE STATISTICS

You can list all available metrics, grouped by the frequency at which they are logged. For each group, you can also view the storage required to store the listed metrics, per day.

Example storage statistics:

```
logged every 60 sec: 61752 bytes or 84.80 Mbytes/day
```

**Procedure**

- To view data storage statistics, enter the following command on your Satellite Server:

```
# less /var/log/pcp/pmlogger/satellite.example.com/pmlogger.log
```

# CHAPTER 5. PCP METRICS

Metrics are stored in a tree-like structure. For example, all network metrics are stored in a node named **network**. Each metric may be a single value, or a list of values, known as instances. For example, kernel load has three instances, a 1-minute, 5-minute, and 15-minute average.

For every metric entry, PCP stores both its data and metadata. This includes the metrics description, data type, units, and dimensions. For example, the metadata enables PCP to output multiple metrics with different dimensions.

If a metric is a counter, its value can only increase. For example, a count of disk write operations on a specific device only increases. When you query the value of a counter metric, PCP converts this into a rate value by default.

In addition to system metrics, such as CPU, memory, kernel, XFS, disk, and network, the following metrics are configured:

| Metric | Description |
|---|---|
| hotproc.* | Basic metrics of key Satellite processes |
| apache.* | Apache HTTP Server metrics |
| postgresql.* | Basic PostgreSQL statistics |
| openmetrics.foreman.fm_rails_* | Satellite metrics |

## 5.1. IDENTIFYING AVAILABLE METRICS

- To list all metrics available through PCP, enter the following command:

```
# pminfo
```

- To list all Satellite metrics and their descriptions, enter the following command:

```
# foreman-rake telemetry:metrics
```

- To list the archived metrics, enter the following command:

```
# less /var/log/pcp/pmlogger/satellite.example.com/pmlogger.log
```

- The **pmlogger** daemon archives data as received, according to its configuration. To retrieve the name of the active log file, enter the following command:

```
# pcp | grep logger
```

  The output includes the file name of the active log file, for example:

```
pmlogger: primary logger: /var/log/pcp/pmlogger/satellite.example.com/20230831.00.10
```

## 5.2. RETRIEVING LIVE METRICS

You can use the PCP CLI tools to retrieve live metrics.

**Procedures**

- To print current values of a particular metric, enter the following command:

```
# pmval -f 1 disk.partitions.write
```

  In this example, metric instances on writes to disk partitions are displayed. PCP converts the number of writes to disk partitions from a counter value to a rate value. The **-f 1** argument specifies to abbreviate the values to one decimal place.

  Example output:

```
metric:    disk.partitions.write
host:      satellite.example.com
semantics: cumulative counter (converting to rate)
units:     count (converting to count / sec)
samples:   all

              vda1             vda2              sr0
               0.0             12.0              0.0
               0.0              1.0              0.0
               0.0              1.0              0.0
               0.0              2.0              0.0
```

- To print system performance summary every 2 seconds, enter the following command:

```
# pmstat -t 2sec
```

## 5.3. RETRIEVING ARCHIVED METRICS

You can use the PCP CLI tools to retrieve metrics from an archive file.

**Examples**

- To list all metrics that were enabled when the archive file was created, enter the following command:

```
# pminfo --archive archive_file
```

- To view the host and time period covered by an archive file, enter the following command:

```
# pmdumplog -l archive_file
```

- To list disk writes for each partition, over the time period covered by the archive file:

```
# pmval --archive /var/log/pcp/pmlogger/satellite.example.com/20230831.00.10 \
  -f 1 disk.partitions.write
```

- To list disk write operations per partition, with a 2-second interval, over the time period between 14:00 and 14:15:

```
# pmval --archive /var/log/pcp/pmlogger/satellite.example.com/20230831.00.10 \
  -d -t 2sec \
  -f 3 disk.partitions.write \
  -S @14:00 -T @14:15
```

- To list average values of all performance metrics, including the time and value of the minimum/maximum, over the time period between 14:00 and 14:30, and format the values as a table:

```
# pmlogsummary /var/log/pcp/pmlogger/satellite.example.com/20230831.00.10 \
  -HlfiImM \
  -S @14:00 \
  -T @14:30 \
  disk.partitions.write \
  mem.freemem
```

- To display system metrics stored in an archive, starting from 14:00, in an interactive manner similar to the **top** tool:

```
# pcp --archive /var/log/pcp/pmlogger/satellite.example.com/20230831.00.10 \
  -S @14:00 \
  atop
```

## 5.4. RETRIEVING METRICS IN THE WEB UI

You can view the Grafana dashboard, which displays widgets with the values of metrics. You can add and remove metrics to suit your requirements. You can also select the time span displayed for each widget.

### Procedure

- Open the following URL in a browser: http://*satellite.example.com*:3000

### Additional resources

- For more details on using Grafana, see the Grafana Labs website.