ORACLE

VM Instance Observability, Monitoring and Alerting within an Oracle Distributed Cloud Environment

A step-by-step guide to deploying an external Grafana Server service to provide a common monitoring and alerting framework across multiple Oracle Distributed Cloud deployments

September, 2023, Version 1.0.1

Purpose statement

This document outlines the steps necessary for using an external Grafana Server service to provide a single, central, and common, observability, monitoring and alerting framework across multiple Oracle Distributed Cloud deployments.

It is intended solely to help you assess the business benefits of using such an approach and to plan your information technology projects accordingly.

Disclaimer

This document in any form, software, or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply.

This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle.

This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Table of contents

List of images

List of tables

Introduction

Oracle Distributed Cloud allows customers to consume Oracle Cloud services anywhere they require.

The Oracle Public Cloud offerings provide Oracle Cloud Infrastructure (OCI) services from multiple locations, or Oracle Public Cloud Regions, around the world.

The Oracle Distributed Cloud portfolio includes Dedicated Region (OCI in the customer's data center), Exadata Cloud@Customer (to provide dedicated cloud-based database services within the customer's own data center) and Oracle Compute Cloud@Customer (OCI compute, networking, and storage services ainthe customer's data center) in a 'connected' mode of operation.

Oracle Private Cloud Appliance (PCA) and the OCI Roving Edge Infrastructure provide identical OCI services but operate in a 'disconnected' mode.

Advantages of the Oracle Compute Cloud@Customer

Oracle Compute Cloud@Customer is fully managed, rack-scale infrastructure that lets organizations consume common OCI services anywhere. Remotely managed by Oracle, it lets customers gain cloud automation and economic benefits, while meeting data residency requirements by controlling their data's location.

Compute Cloud@Customer systems can be paired with Oracle Exadata to create an ideal infrastructure for scalable, multitier applications.

Advantages of Oracle Private Cloud Appliance

Oracle Private Cloud Appliance (PCA) is an Oracle Engineered System designed for implementing the application and middleware tiers. PCA is an integrated hardware and software system that reduces infrastructure complexity and deployment time for virtualized workloads in private clouds. It is a complete platform for a wide range of application types and workloads, with built-in management, compute, storage, and networking resources. PCA provides excellent performance and other system properties for hosting a broad range of applications.

Oracle Private Cloud Appliance X9-2 is the latest member of the Oracle Private Cloud Appliance product family. PCA provides cloud and administrative services for a supporting range of workloads including cloud native applications. It makes use of a modern microservices architecture, Kubernetes, and related technologies, for a future-proofed software stack.

A key new feature of Oracle Private Cloud Appliance X9-2, compared to previous versions, is that it delivers private cloud infrastructure and architecture consistent with Oracle Cloud Infrastructure (OCI). Oracle Private Cloud Appliance brings APIs and SDKs compatible with Oracle Cloud Infrastructure (OCI) to an on-premises implementation at rack scale, making workloads, user experience, tool sets and skills portable between private and public clouds. Oracle Private Cloud Appliance can be paired with Oracle Exadata to create an ideal infrastructure for scalable, multi-tier applications. Customers preferring or requiring an on-premises solution can realize the operational benefits of public cloud deployments using Oracle Private Cloud Appliance X9-2.

Advantages of OCI Roving Edge Infrastructure

Oracle Cloud Infrastructure (OCI) Roving Edge Infrastructure accelerates deployment of cloud workloads outside the data center. Ruggedized devices deliver cloud computing and storage services at the edge of networks in disconnected locations, allowing faster processing close to the data source and enabling faster insights into the data.

Roving Edge devices provide faster processing close to the data source enabling faster insights into the data collected. Leverage Roving Edge Infrastructure devices with powerful computing capabilities for ingesting and processing large amounts of streaming data from sensors in remote locations. Enable seamless deployment of applications for organizations such as embassies and consulates, government offices, forward operating bases, and remote campuses. Use built-in GPUs or attached VPU/TPU accelerators for faster processing of AI and ML workloads without relying on network connectivity to Oracle Cloud Infrastructure.

Existing OCI compute images and object storage can be synchronized to Oracle Roving Edge devices using the same portal and tenancy tooling as our public regions.

Monitoring & Alerting within an Oracle Distributed Cloud Environment

Multiple Monitoring & Alerting options are available.

The Compute Cloud@Customer operates in a fully 'connected' mode, with a permanent communications channel back to the parent OCI Region. Use of the standard Oracle Cloud Agent can provide centralized monitoring & alerting through the parent OCI Region Oracle Management Cloud services. The use of alternative tools for the collection, collation, and reporting of VM Instance resource utilization is also possible.

The Private Cloud Appliance X9-2, operating in a 'disconnected' mode, provides monitoring and alerting capabilities through a fully integrated Grafana service. Each Private Cloud Appliance maintains its own, independent Grafana service. Use of the OCI Oracle Cloud Agents can be considered but requires additional services to be made available.

The OCI Roving Edge Infrastructure, like the Private Cloud Appliance, operates in a 'disconnected' mode.

For customers with multiple Oracle Distributed Cloud deployments, monitoring & alerting services can be consolidated to monitor each Oracle Distributed Cloud deployment through a single, centralized, external Grafana instance.

For customers who are not using Grafana widely within their organization, a new Grafana instance is likely to be needed to act as the centralized "Single Window" into their Oracle Distributed Cloud environments.

Scope and content

This document describes how Virtual Machine instances across multiple Oracle Distributed Cloud deployments can be configured to provide system resource metrics for observability, monitoring and alerting purposes to a single, central, external Grafana Server service.

Detailed step-by-step guides are provided to cover two key areas:

- The installation & configuration of services within the VM Instances
- The installation & configuration of services within the eternal Grafana Server service

Each area is covered in a specific section below, along with a description of the tools used for the collection, collation, and visualization of the VM Instance system metrics captured.

Performance Co-Pilot

Performance Co-Pilot (https://pcp.io/) is a lightweight systems performance toolkit providing an extensible framework to collect, collate and report on a wide number of system metrics.

This toolkit consists of several modules to provide both *COLLECTION* and *MONITORING* capabilities for a wide range of systems level performance & utilization metrics.

The basic PCP package (pcp) provide the following tools:

- pmcd - performance metrics collector daemon
- pmstat – Outputs an ASCII high-level summary of system performance.
- pmie – An inference engine that can evaluate predicate-action rules to perform alarms and automate system management tasks.
- pminfo – Interrogate specific performance metrics and the metadata that describes them.
- pmlogger – Generates PCP archives of performance metrics suitable for replay by most PCP tools.
- pmproxy - proxy for performance metrics collector and querying
- pmrep – Highly customizable performance metrics reporter with support for various different output modes.
- pmval – Simple periodic reporting for some or all instances of a performance metric, with optional VCR time control.

If the PCP GUI package is installed (pcp-gui) then the following additional tools are available:

- pmchart – Displays trends over time of arbitrarily selected performance metrics from one or more hosts.
- pmdumptext – Produce ASCII reports for arbitrary combinations of performance metrics
- pmsnap – Generates performance summary snapshot images
- pmtime – Time control utility for coordinating the time between multiple tools (including pmchart and pmval).

If the PCP System Tools package is installed (pcp-system-tools) then the following additional tools are available:

- dstat – a general performance analysis tool allowing views of multiple system resources instantly
- pcp2csv – a CSV extraction tool for PCP Archive Log files
- pmiostat – reports I/O statistics for scsi devices (by default) or device-mapper devices (if the -x dm option is specified)
- pmrep – customizable performance metrics reporting tool
- pmrepconf – manages the local configuration for pmrep

The PCP Doc package (pcp-doc) provides additional documentation and examples locally to the VM Instance within which it has been installed.

The PCP Cockpit package (cockpit-pcp) provides integration between the PCP tools and the Cockpit Web UI for Linux Systems Administration.

The PCP toolkit is available for the following Operating Systems:

- Oracle Linux
- Red Hat Linux
- Ubuntu
- Debian
- SUSE Linux
- Oracle Solaris
- Mac OSX
- Microsoft Windows

Additional 'plugin agents' can be used to extend this toolkit to collect metrics for:

- Databases
- Web Servers
- Cluster infrastructure
- Mail systems
- Cisco routers

Details about the capabilities of the Performance Co-Pilot (pcp) framework is well documented. The following URL provides further information on the capabilities for Performance Co-Pilot - https://pcp.readthedocs.io/en/latest/

Each area will now be described in more detail.

PCP Collection Modules

The following components are available to be installed for the collection of systems performance metrics.

Performance Metrics Collector Daemon

The Performance Metrics Collector Daemon (pmcd) is the collector used by the Performance Co-Pilot (PCP) framework to gather performance metrics on a system. As a rule, there must be an instance of pmcd running on a system for any performance metrics to be available to the PCP toolkit.

Performance Metrics Domain Agent(s)

The PCP architecture is distributed in the sense that any PCP tool may be executing remotely. On the host (or hosts) being monitored, each domain of performance metrics, whether the kernel, a service layer, a database management system, a web server, an application, etc. requires a Performance Metrics Domain Agent (pmda) which is responsible for collecting performance measurements from that domain. All PMDAs are controlled by the Performance Metrics Collector Daemon (pmcd) on the same host.

PCP Archive Logger (pmlogger)

The PCP Archive Logger (pmlogger) creates the archive logs of performance metric values that may be 'played back' by other Performance Co-Pilot tools. These logs form the basis of retrospective performance analysis services common to the PCP toolkit.

Performance Metrics Proxy (pmproxy)

The Performance Metrics Proxy(pmproxy) acts as a protocol proxy, allowing Performance Co-Pilot (PCP) monitoring clients to connect to one or more pmcd and/or redis-server instances via pmproxy.

pmproxy can be deployed in a firewall domain, or on a cluster 'head' node where the IP (Internet Protocol) address of the hosts where pmcd and/or redis-server are running may be unknown to the PCP monitoring clients, but where the IP address of the host running pmproxy is known to these clients. Similarly, the clients may have network connectivity only to the host where pmproxy is running, while there is network connectivity from that host to the hosts of interest where pmcd and/or redis-server are running.

PCP Monitoring Modules

Over seventy PCP Monitoring Modules (pmda's) are available.

Some Performance Metrics Domain Agents (pmda(s)) shipped with Performance Co-Pilot (PCP) are designed to be installed and activated on every collector host, for example, linux, windows, darwin, pmcd, and process PMDAs.

Other PMDAs are designed for optional activation and require some user action to make them operational. In some cases, these PMDAs expect local site customization to reflect the operational environment, the system configuration, or the production workload. This customization is typically supported by interactive installation scripts for each PMDA.

Each PMDA has its own directory located below ${PCP_PMDAS_DIR}. Each directory contains a Remove script to unconfigure the PMDA, remove the associated metrics from the PMNS, and restart the pmcd daemon; and an Install script to install the PMDA, update the PMNS, and restart the PMCD daemon.

PCP and Oracle Linux

The Performance Co-Pilot packages have been included within the base Oracle Linux distributions since Oracle Linux 6. The version of PCP provided depends on the Oracle Linux version and whether non-security updates are still being provided.

Oracle Linux PCP versions

The table below shows the PCP version included within each Oracle Linux version:

ORACLE LINUX VERSION	PERFORMANCE CO-PILOT VERSION	ULN CHANNEL
Oracle Linux 6	3.10.9-9.0.1.el6	ol6_latest
Oracle Linux 7	4.3.2-13.0.13.el7_9	ol7_latest
Oracle Linux 8	5.3.7-16.0.1.el8	ol8_appstream
Oracle Linux 9	6.0.1-4.0.2.el9	ol9_appstream

Oracle Linux – Performance Co-Pilot version matrix

Oracle Linux PCP available PMDA's

The table below shows the PMDA modules available for each Oracle Linux version:

PMDA MODULE	ORACLE LINUX 6	ORACLE LINUX 7	ORACLE LINUX 8	ORACLE LINUX 9
activemq	Y	Y	Y	Y
apache	Y	Y	Y	Y
bash	Y	Y	Y	Y
bcc		Y	Y	Y
bind2		Y	Y	Y
bonding	Y	Y	Y	Y
bpf				Y
bpftrace			Y	Y
cifs	Y		Y	Y
cisco	Y	Y	Y	Y
dbping	Y	Y	Y	Y
denki			Y	Y
dm	Y	Y	Y	Y
docker		Y	Y	Y
ds389	Y	Y	Y	Y
ds389log	Y	Y	Y	Y
elasticsearch	Y	Y	Y	Y
gfs2	Y	Y	Y	Y
gluster	Y	Y	Y	Y
gpfs	Y	Y	Y	Y
gpsd	Y	Y	Y	Y
hacluster			Y	Y
haproxy		Y	Y	Y
infiniband	Y		Y	Y

PMDA MODULE	ORACLE LINUX 6	ORACLE LINUX 7	ORACLE LINUX 8	ORACLE LINUX 9
json		Y	Y	Y
kvm	Y	Y	Y	Y
libvirt		Y	Y	Y
lio		Y	Y	Y
lmsensors	Y	Y	Y	Y
logger	Y	Y	Y	Y
lustre	Y	Y	Y	Y
lustrecomm	Y	Y	Y	Y
mailq	Y	Y	Y	Y
memcache	Y	Y	Y	Y
mic	Y		Y	Y
mongodb			Y	Y
mounts	Y	Y	Y	Y
mssql			Y	Y
mysql	Y	Y	Y	Y
named	Y	Y	Y	Y
netcheck			Y	Y
netfilter	Y	Y	Y	Y
news	Y	Y	Y	Y
nfsclient	Y	Y	Y	Y
nginx	Y	Y	Y	Y
nvidia-gpu	Y	Y	Y	Y
openmetrics			Y	Y
openvswitch			Y	Y
oracle			Y	Y
papi	Y		Y	
pdns	Y	Y	Y	Y
perfevent			Y	Y
podman			Y	Y
postfix	Y	Y	Y	Y
postgresql	Y	Y	Y	Y
prometheus		Y	Y	
rabbitmq			Y	Y
redis		Y	Y	Y
roomtemp	Y	Y	Y	Y
rpm	Y	Y	Y	
rsyslog	Y		Y	Y
samba	Y		Y	Y

PMDA MODULE	ORACLE LINUX 6	ORACLE LINUX 7	ORACLE LINUX 8	ORACLE LINUX 9
sendmail	Y	Y	Y	Y
shping	Y	Y	Y	Y
slurm	Y		Y	Y
smart			Y	Y
snmp	Y		Y	Y
sockets			Y	Y
statsd			Y	Y
summary	Y	Y	Y	Y
systemd			Y	Y
trace	Y	Y	Y	Y
unbound	Y	Y	Y	Y
weblog	Y	Y	Y	Y
zimbra	Y		Y	Y
zswap	Y	Y	Y	Y

Oracle Linux – Performance Co-Pilot Performance Metrics Domain Agents

Over seventy PMDA modules are available within each of the publicly available Oracle Linux Channels.

Installing & Configuring PCP within a VM Instance

The following section provides a step-by-step guide for the installation and configuration of the basic PCP tools required to collect and collate local resource utilization metrics within a VM Instance.

As a bare minimum, the Performance Metrics Collection Daemon (pmcd) and PCP Archive Logger (pmlogger) will need to be configured.

The Performance Metrics Proxy (pmproxy) is only required to be enabled when direct access between the VM Instance and end-point collector requires routing through firewalls etc.

Additional Performance Metrics Domain Agents (PMDAs) can be added to the base installation to enable metrics collection for any specific services installed within the VM Instance.

Further, optional, PCP components, such as pcp-gui, pcp-doc, pcp-system-tools & cockpit-pcp only need be installed where required.

Basic PCP Installation

With this example, the basic system level metrics will be collected using the 'pmcd' service, and locally retained and archived using the 'pmlogger' services.

For the purposes of this document, the ‘pcp-zeroconf’ RPM package will be used to provide a minimal installation footprint.

pcp-zeroconf is intended to simplify the installation and configuration of the most commonly needed PCP features in customer support environments, where the need for comprehensive long-term logging of system level performance data is combined with minimal overheads.

The pcp-zeroconf RPM package installs a minimal set of dependencies for the server-side data collection and does not include client tools. The package has been part of PCP from version 3.11.10 onwards and is also included within Oracle Linux from release 7.4 and later.

As the name suggests, there are no further installation or configurations steps required in order to collect performance data archives.

NOTE:

It is important to make sure there is sufficient space on the /var filesystem (or root filesystem if /var is not a sub mount) for on-going log collection

- approximately 5GB will be required on most VM Instances.
- On larger VM Instances, at least 10GB of space on /var will be used on an on-going basis.

By default, PCP archive logs are automatically culled after 14 days – see the man pages for pmlogger_daily, or pcp documentation, for instructions on changing this if needed.

The code block below shows the command line output from this installation:

```
[root@brm-pcapm-pcp-01 ~]# dnf install pcp-zeroconf
last metadata expiration check: 10:48:20 ago on fri 28 jul 2023 17:49:35 bst.
dependencies resolved.
================================================================================================================================================
 package                                architecture             version                                repository
================================================================================================================================================
installing:
 pcp-zeroconf                           x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
installing dependencies:
 pcp                                    x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-conf                               x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-doc                                noarch                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-libs                               x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-pmda-dm                            x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-pmda-nfsclient                     x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-pmda-openmetrics                   x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-selinux                            x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 pcp-system-tools                       x86_64                   6.0.1-4.0.2.el9                        ol9_appstream
 python3-pcp                            x86_64                   6.0.1-4.0.2.el9                        ol9_appstream

transaction summary
================================================================================================================================================
install  11 packages

total download size: 6.7 m
installed size: 16 m
is this ok [y/n]: y
downloading packages:
(1/11): pcp-conf-6.0.1-4.0.2.el9.x86_64.rpm
(2/11): pcp-libs-6.0.1-4.0.2.el9.x86_64.rpm
(3/11): pcp-doc-6.0.1-4.0.2.el9.noarch.rpm
(4/11): pcp-pmda-dm-6.0.1-4.0.2.el9.x86_64.rpm
(5/11): pcp-pmda-nfsclient-6.0.1-4.0.2.el9.x86_64.rpm
(6/11): pcp-6.0.1-4.0.2.el9.x86_64.rpm
(7/11): pcp-selinux-6.0.1-4.0.2.el9.x86_64.rpm
(8/11): pcp-pmda-openmetrics-6.0.1-4.0.2.el9.x86_64.rpm
(9/11): pcp-zeroconf-6.0.1-4.0.2.el9.x86_64.rpm
(10/11): pcp-system-tools-6.0.1-4.0.2.el9.x86_64.rpm
(11/11): python3-pcp-6.0.1-4.0.2.el9.x86_64.rpm
------------------------------------------------------------------------------------------------------------------------------------------------
total
running transaction check
transaction check succeeded.
running transaction test
transaction test succeeded.
running transaction
  preparing        :
  running scriptlet: pcp-selinux-6.0.1-4.0.2.el9.x86_64
  installing       : pcp-selinux-6.0.1-4.0.2.el9.x86_64
  running scriptlet: pcp-selinux-6.0.1-4.0.2.el9.x86_64
libsemanage.semanage_direct_install_info: overriding pcp module at lower priority 100 with module at priority 200.

  installing       : pcp-doc-6.0.1-4.0.2.el9.noarch
  installing       : pcp-conf-6.0.1-4.0.2.el9.x86_64
  installing       : pcp-libs-6.0.1-4.0.2.el9.x86_64
  running scriptlet: pcp-6.0.1-4.0.2.el9.x86_64
creating group 'pcp' with gid 977.
creating user 'pcp' (performance co-pilot) with uid 977 and gid 977.

  installing       : pcp-6.0.1-4.0.2.el9.x86_64
  running scriptlet: pcp-6.0.1-4.0.2.el9.x86_64
  installing       : python3-pcp-6.0.1-4.0.2.el9.x86_64
  installing       : pcp-pmda-nfsclient-6.0.1-4.0.2.el9.x86_64
  installing       : pcp-pmda-openmetrics-6.0.1-4.0.2.el9.x86_64
  installing       : pcp-system-tools-6.0.1-4.0.2.el9.x86_64
  installing       : pcp-pmda-dm-6.0.1-4.0.2.el9.x86_64
  installing       : pcp-zeroconf-6.0.1-4.0.2.el9.x86_64
  running scriptlet: pcp-zeroconf-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-conf-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-doc-6.0.1-4.0.2.el9.noarch
  verifying        : pcp-libs-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-pmda-dm-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-pmda-nfsclient-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-pmda-openmetrics-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-selinux-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-system-tools-6.0.1-4.0.2.el9.x86_64
  verifying        : pcp-zeroconf-6.0.1-4.0.2.el9.x86_64
  verifying        : python3-pcp-6.0.1-4.0.2.el9.x86_64

installed:
  pcp-6.0.1-4.0.2.el9.x86_64                  pcp-conf-6.0.1-4.0.2.el9.x86_64      pcp-doc-6.0.1-4.0.2.el9.noarch
  pcp-libs-6.0.1-4.0.2.el9.x86_64             pcp-pmda-dm-6.0.1-4.0.2.el9.x86_64   pcp-pmda-nfsclient-6.0.1-4.0.2.el9.x86_64
  pcp-pmda-openmetrics-6.0.1-4.0.2.el9.x86_64 pcp-selinux-6.0.1-4.0.2.el9.x86_64   pcp-system-tools-6.0.1-4.0.2.el9.x86_64
  pcp-zeroconf-6.0.1-4.0.2.el9.x86_64         python3-pcp-6.0.1-4.0.2.el9.x86_64
complete!
[root@brm-pcapm-pcp-01 ~]#
```

Performance Co-Pilot Basic Installation

Finally, confirm that the base Performance Co-Pilot services are enabled and running:

```
[root@brm-pcapm-pcp-01 ~]# systemctl status pmcd pmlogger pmie
● pmcd.service - performance metrics collector daemon
     loaded: loaded (/usr/lib/systemd/system/pmcd.service; enabled; preset: disabled)
     active: active (running) since sat 2023-07-29 05:15:19 bst; 6min ago
       docs: man:pmcd(1)
   main pid: 7921 (pmcd)
      tasks: 9 (limit: 98963)
     memory: 38.5m
        cpu: 2.068s
     cgroup: /system.slice/pmcd.service
             ├─7921 /usr/libexec/pcp/bin/pmcd -i 10.80.160.48 -a
             ├─7925 /var/lib/pcp/pmdas/root/pmdaroot
             ├─7926 /var/lib/pcp/pmdas/proc/pmdaproc -d 3
             ├─7928 /var/lib/pcp/pmdas/xfs/pmdaxfs -d 11
             ├─7929 /var/lib/pcp/pmdas/linux/pmdalinux
             ├─7930 python3 /var/lib/pcp/pmdas/nfsclient/pmdanfsclient.python
             ├─7934 /var/lib/pcp/pmdas/kvm/pmdakvm -d 95
             ├─7935 /var/lib/pcp/pmdas/dm/pmdadm -d 129
             └─7936 python3 /var/lib/pcp/pmdas/openmetrics/pmdaopenmetrics.python

jul 29 05:15:18 brm-pcapm-pcp-01.us.oracle.com systemd[1]: starting performance metrics collector daemon...
jul 29 05:15:19 brm-pcapm-pcp-01.us.oracle.com systemd[1]: started performance metrics collector daemon.

● pmlogger.service - performance metrics archive logger
     loaded: loaded (/usr/lib/systemd/system/pmlogger.service; enabled; preset: disabled)
     active: active (running) since sat 2023-07-29 04:39:30 bst; 42min ago
       docs: man:pmlogger(1)
   main pid: 3702 (pmlogger)
      tasks: 1 (limit: 98963)
     memory: 17.8m
        cpu: 2.612s
     cgroup: /system.slice/pmlogger.service
             └─3702 /usr/libexec/pcp/bin/pmlogger -n -p -r -t24h10m -c config.default -v 100mb -m pmlogger_check %y%m%d.%h.%m

jul 29 04:39:08 brm-pcapm-pcp-01.us.oracle.com systemd[1]: starting performance metrics archive logger...
jul 29 04:39:10 brm-pcapm-pcp-01.us.oracle.com pmlogger[2909]: /usr/libexec/pcp/lib/pmlogger: warning: performance co-pilot
archive logger(s) not permanently enabled.
jul 29 04:39:10 brm-pcapm-pcp-01.us.oracle.com pmlogger[2909]:     to enable pmlogger, run the following as root:
jul 29 04:39:10 brm-pcapm-pcp-01.us.oracle.com pmlogger[2909]:     # /usr/bin/systemctl enable pmlogger.service
jul 29 04:39:30 brm-pcapm-pcp-01.us.oracle.com systemd[1]: started performance metrics archive logger.

● pmie.service - performance metrics inference engine
     loaded: loaded (/usr/lib/systemd/system/pmie.service; enabled; preset: disabled)
     active: active (running) since sat 2023-07-29 04:39:22 bst; 42min ago
       docs: man:pmie(1)
   main pid: 3495 (pmie)
      tasks: 1 (limit: 98963)
     memory: 1.4m
        cpu: 1.916s
     cgroup: /system.slice/pmie.service
             └─3495 /usr/bin/pmie -b -f -p -l /var/log/pcp/pmie/brm-pcapm-pcp-01.us.oracle.com/pmie.log -c config.default

jul 29 04:39:08 brm-pcapm-pcp-01.us.oracle.com systemd[1]: starting performance metrics inference engine...
jul 29 04:39:10 brm-pcapm-pcp-01.us.oracle.com pmie[2908]: /usr/libexec/pcp/lib/pmie: warning: performance co-pilot inference
engine (pmie) not permanently enabled.
jul 29 04:39:10 brm-pcapm-pcp-01.us.oracle.com pmie[2908]:     to enable pmie, run the following as root:
jul 29 04:39:10 brm-pcapm-pcp-01.us.oracle.com pmie[2908]:     # /usr/bin/systemctl enable pmie.service
jul 29 04:39:22 brm-pcapm-pcp-01.us.oracle.com systemd[1]: started performance metrics inference engine.
[root@brm-pcapm-pcp-01 ~]#
```

Performance Co-Pilot Basic Services Status

The base services for Performance Co-Pilot, pmcd, pmlogger, and pmie, are all running correctly.

Basic PCP Configuration

Although the pcp-zeroconf RPM has installed and started the base PCP services, some final configuration changes are still required to ensure that the services restart on reboot and to enable external collation of each VM Instance's PCP performance metrics.

The following changes now need to be made.

Enable PCP base services

Using the 'systemctl' command, enable the three base services to restart on reboot:

```
[root@brm-pcapm-pcp-01 ~]# systemctl enable pmcd pmlogger pmie
[root@brm-pcapm-pcp-01 ~]#
```

Performance Co-Pilot Enable Base Services

Network Interface change for pmcd

Append the following text '-i <system IP address>' to the '/etc/pcp/pmcd/pmcd.options' configuration file to enable the pmcd service to be addressable by an external collector system.

Firewall Changes

Configure the default firewall services to permit the pmcd daemon to be visible to the external collation service:

```
[root@brm-pcapm-pcp-01 ~]# firewall-cmd --add-service=pmcd --permanent
success
[root@brm-pcapm-pcp-01 ~]# firewall-cmd --reload
success
[root@brm-pcapm-pcp-01 ]#
```

Performance Co-Pilot Enable PMCD Firewall Access

Restart PCP base services and check status

Finally, restart the base PCP services and recheck the status:

```
[root@brm-pcapm-pcp-01 ~]# systemctl restart pmcd pmlogger pmie
[root@brm-pcapm-pcp-01 ~]# systemctl status pmcd pmlogger pmie
● pmcd.service - performance metrics collector daemon
     loaded: loaded (/usr/lib/systemd/system/pmcd.service; enabled; preset: disabled)
     active: active (running) since sat 2023-07-29 05:56:07 bst; 8s ago
       docs: man:pmcd(1)
   main pid: 11145 (pmcd)
      tasks: 9 (limit: 98963)
     memory: 39.4m
        cpu: 1.563s
     cgroup: /system.slice/pmcd.service
             ├─11145 /usr/libexec/pcp/bin/pmcd -i 10.80.160.48 -a
             ├─11149 /var/lib/pcp/pmdas/root/pmdaroot
             ├─11150 /var/lib/pcp/pmdas/proc/pmdaproc -d 3
             ├─11152 /var/lib/pcp/pmdas/xfs/pmdaxfs -d 11
             ├─11153 /var/lib/pcp/pmdas/linux/pmdalinux
             ├─11154 python3 /var/lib/pcp/pmdas/nfsclient/pmdanfsclient.python
             ├─11158 /var/lib/pcp/pmdas/kvm/pmdakvm -d 95
             ├─11159 /var/lib/pcp/pmdas/dm/pmdadm -d 129
             └─11160 python3 /var/lib/pcp/pmdas/openmetrics/pmdaopenmetrics.python

jul 29 05:56:06 brm-pcapm-pcp-01.us.oracle.com systemd[1]: starting performance metrics collector daemon...
jul 29 05:56:07 brm-pcapm-pcp-01.us.oracle.com systemd[1]: started performance metrics collector daemon.

● pmlogger.service - performance metrics archive logger
     loaded: loaded (/usr/lib/systemd/system/pmlogger.service; enabled; preset: disabled)
     active: active (running) since sat 2023-07-29 05:56:08 bst; 7s ago
       docs: man:pmlogger(1)
   main pid: 11700 (pmlogger)
      tasks: 1 (limit: 98963)
     memory: 2.9m
        cpu: 1.251s
     cgroup: /system.slice/pmlogger.service
             └─11700 /usr/libexec/pcp/bin/pmlogger -n -p -r -t24h10m -c config.default -v 100mb -m pmlogger_check %y%m%d.%h.%m

jul 29 05:56:07 brm-pcapm-pcp-01.us.oracle.com systemd[1]: starting performance metrics archive logger...
jul 29 05:56:08 brm-pcapm-pcp-01.us.oracle.com systemd[1]: started performance metrics archive logger.

● pmie.service - performance metrics inference engine
     loaded: loaded (/usr/lib/systemd/system/pmie.service; enabled; preset: disabled)
     active: active (running) since sat 2023-07-29 05:56:08 bst; 8s ago
       docs: man:pmie(1)
   main pid: 11650 (pmie)
      tasks: 1 (limit: 98963)
     memory: 1.4m
        cpu: 598ms
     cgroup: /system.slice/pmie.service
             └─11650 /usr/bin/pmie -b -f -p -l /var/log/pcp/pmie/brm-pcapm-pcp-01.us.oracle.com/pmie.log -c config.default

jul 29 05:56:07 brm-pcapm-pcp-01.us.oracle.com systemd[1]: starting performance metrics inference engine...
jul 29 05:56:08 brm-pcapm-pcp-01.us.oracle.com systemd[1]: started performance metrics inference engine.
[root@brm-pcapm-pcp-01 ~]#
```

Performance Co-Pilot Basic Services Status - updated

All services are Active and Enabled.

Basic PCP Status Checks

Having completed the basic installation and additional configuration services a few simple checks can be run to confirm that performance metrics are being collected. The command 'pcp' will show the detailed status of the pmcd service:

```
[root@brm-pcapm-pcp-01 ~]# pcp
performance co-pilot configuration on brm-pcapm-pcp-01.us.oracle.com:

 platform: linux brm-pcapm-pcp-01.us.oracle.com 5.15.0-103.114.4.el9uek.x86_64 #2 smp mon jun 26 10:09:23 pdt 2023 x86_64
 hardware: 2 cpus, 1 disk, 1 node, 15525mb ram
 timezone: bst-1
 services: pmcd
     pmcd: version 6.0.1-4, 12 agents, 6 clients
     pmda: root pmcd proc pmproxy xfs linux nfsclient mmv kvm jbd2
           dm openmetrics
 pmlogger: primary logger: /var/log/pcp/pmlogger/brm-pcapm-pcp-01.us.oracle.com/20230729.05.56
     pmie: primary engine: /var/log/pcp/pmie/brm-pcapm-pcp-01.us.oracle.com/pmie.log
[root@brm-pcapm-pcp-01 ~]#
```

Performance Co-Pilot Basic Installation PMCD Details

This shows that the Performance Metrics Collector Daemon (pmcd) has twelve associated Performance Metrics Domain Agents (pmda) associated with the service and that the pcp logger service 'pmlogger' is active and recording metrics into the '/var/log/pcp/pmlogger' directory.

The following PDMA modules are reported as installed and active:

- root
- pmcd
- proc
- pmproxy
- xfs
- linux
- nfsclient
- mmv
- kvm
- jbd2
- dm
- openmetrics

The command 'pminfo -t' will list the performance metrics (almost 3,500) being collected and that are available for further analysis.

Please see the Performance Co-Pilot documentation for further information with regards additional PMDA agents that can be enabled.

Extend PCP services to multiple VM Instances

Having completed the installation and configuration for the initial VM Instance, repeat as necessary across the remaining VM Instances within Oracle Distributed Cloud estate as required.

Section References

The following URL's provide links to additional documentation:

- Oracle Linux 9 – Reference Library – https://docs.oracle.com/en/operating-systems/oracle-linux/9/
- Oracle Linux 9 – Working with Performance Co-Pilot - https://docs.oracle.com/en/operating-systems/oracle-linux/9/monitoring/monitoring-WorkingWithPerformanceCoPilot.html#pcp
- Oracle Linux 8 – Reference Library – https://docs.oracle.com/en/operating-systems/oracle-linux/8/
- Oracle Linux 8 – Working with Performance Co-Pilot – https://docs.oracle.com/en/operating-systems/oracle-linux/8/monitoring/monitoring-WorkingWithPerformanceCoPilot.html#pcp
- Performance Co-Pilot documentation – https://pcp.readthedocs.io/en/latest/
- Performance Co-Pilot documentation – Quick Reference Guide – https://pcp.readthedocs.io/en/latest/QG/QuickReferenceGuide.html

Collection & Collation of PCP metrics for multiple VM Instances

The previous section of this document provided instructions for the initial installation and configuration of Performance Co-Pilot metric collection services for individual VM Instances.

A VM Instance will now have the full Performance Co-Pilot toolset installed and configured to collect PCP Archive logs for both itself, and remote hosts.

Advanced PCP Installation

As before the Performance Co-Pilot tools will need to be installed.

The code block below shows the command line output from this installation:

```
[root@brm-pcapm-pcp-02 ~]# dnf install pcp pcp-system-tools pcp-gui pcp-doc
Last metadata expiration check: 15:12:03 ago on Fri 28 Jul 2023 17:49:35 BST.
Dependencies resolved.
================================================================================================================================================
 Package                 Architecture                Version                       Repository                                Size
================================================================================================================================================
Installing:
 pcp                     x86_64                      6.0.1-4.0.2.el9               ol9_appstream                            1.7 M
 pcp-doc                 noarch                      6.0.1-4.0.2.el9               ol9_appstream                            3.6 M
 pcp-gui                 x86_64                      6.0.1-4.0.2.el9               Ol9_appstream                            847 k
 pcp-system-tools        x86_64                      6.0.1-4.0.2.el9               ol9_appstream                            385 k
Installing dependencies:
 pcp-conf                x86_64                      6.0.1-4.0.2.el9               ol9_appstream                             37 k
 pcp-libs                x86_64                      6.0.1-4.0.2.el9               ol9_appstream                            634 k
 pcp-selinux             x86_64                      6.0.1-4.0.2.el9               ol9_appstream                             30 k
 pcre2-utf16             x86_64                      10.40-2.0.2.el9               ol9_appstream                            214 k
 python3-pcp             x86_64                      6.0.1-4.0.2.el9               ol9_appstream                            189 k
 qt5-qtbase              X86_64                      5.15.3-1.el9                  ol9_appstream                            3.7 M
 qt5-qtbase-common       noarch                      5.15.3-1.el9                  ol9_appstream                             12 k
 qt5-qtbase-gui          x86_64                      5.15.3-1.el9                  ol9_appstream                            6.4 M
 qt5-qtsvg               x86_64                      5.15.3-1.el9                  ol9_appstream                            199 k
 xcb-util-image          x86_64                      0.4.0-19.el9                  ol9_appstream                             20 k
 xcb-util-keysyms        x86_64                      0.4.0-17.el9                  ol9_appstream                             15 k
 xcb-util-renderutil     x86_64                      0.3.9-20.el9                  ol9_appstream                             18 k
 xcb-util-wm             x86_64                      0.4.1-22.el9                  ol9_appstream                             32 k

Transaction Summary
================================================================================================================================================
Install  17 Packages

Total download size: 18 M
Installed size: 49 M
Is this ok [y/N]: y
Downloading Packages:
(1/17): pcp-conf-6.0.1-4.0.2.el9.x86_64.rpm                                                    52 kB/s |  37 kB     00:00
(2/17): pcp-gui-6.0.1-4.0.2.el9.x86_64.rpm                                                    1.4 MB/s | 847 kB     00:00
(3/17): pcp-doc-6.0.1-4.0.2.el9.noarch.rpm                                                    2.4 MB/s | 3.6 MB     00:01
(4/17): pcp-libs-6.0.1-4.0.2.el9.x86_64.rpm                                                   2.4 MB/s | 634 kB     00:00
(5/17): pcp-selinux-6.0.1-4.0.2.el9.x86_64.rpm                                                432 kB/s |  30 kB     00:00
(6/17): pcre2-utf16-10.40-2.0.2.el9.x86_64.rpm                                                1.2 MB/s | 214 kB     00:00
(7/17): pcp-system-tools-6.0.1-4.0.2.el9.x86_64.rpm                                           1.5 MB/s | 385 kB     00:00
(8/17): python3-pcp-6.0.1-4.0.2.el9.x86_64.rpm                                                1.8 MB/s | 189 kB     00:00
(9/17): qt5-qtbase-common-5.15.3-1.el9.noarch.rpm                                              29 kB/s |  12 kB     00:00
(10/17): qt5-qtbase-5.15.3-1.el9.x86_64.rpm                                                   2.8 MB/s | 3.7 MB     00:01
(11/17): pcp-6.0.1-4.0.2.el9.x86_64.rpm                                                       547 kB/s | 1.7 MB     00:03
(12/17): qt5-qtsvg-5.15.3-1.el9.x86_64.rpm                                                    647 kB/s | 199 kB     00:00
(13/17): xcb-util-image-0.4.0-19.el9.x86_64.rpm                                                85 kB/s |  20 kB     00:00
(14/17): xcb-util-renderutil-0.3.9-20.el9.x86_64.rpm                                          261 kB/s |  18 kB     00:00
(15/17): qt5-qtbase-gui-5.15.3-1.el9.x86_64.rpm                                               3.2 MB/s | 6.4 MB     00:01
(16/17): xcb-util-wm-0.4.1-22.el9.x86_64.rpm                                                   45 kB/s |  32 kB     00:00
(17/17): xcb-util-keysyms-0.4.0-17.el9.x86_64.rpm                                             9.8 kB/s |  15 kB     00:01
------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                         3.6 MB/s |  18 MB     00:04
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                              1/1
  Installing       : xcb-util-wm-0.4.1-22.el9.x86_64                                                                             1/17
  Installing       : xcb-util-renderutil-0.3.9-20.el9.x86_64                                                                     2/17
  Installing       : xcb-util-keysyms-0.4.0-17.el9.x86_64                                                                        3/17
  Installing       : xcb-util-image-0.4.0-19.el9.x86_64                                                                          4/17
  Installing       : pcre2-utf16-10.40-2.0.2.el9.x86_64                                                                          5/17
  Installing       : qt5-qtbase-common-5.15.3-1.el9.noarch                                                                       6/17
  Running scriptlet: qt5-qtbase-5.15.3-1.el9.x86_64                                                                              7/17
  Installing       : qt5-qtbase-5.15.3-1.el9.x86_64                                                                              7/17
  Running scriptlet: qt5-qtbase-5.15.3-1.el9.x86_64                                                                              7/17
  Installing       : qt5-qtbase-gui-5.15.3-1.el9.x86_64                                                                          8/17
  Installing       : qt5-qtsvg-5.15.3-1.el9.x86_64                                                                               9/17
  Running scriptlet: pcp-selinux-6.0.1-4.0.2.el9.x86_64                                                                         10/17
  Installing       : pcp-selinux-6.0.1-4.0.2.el9.x86_64                                                                         10/17
  Running scriptlet: pcp-selinux-6.0.1-4.0.2.el9.x86_64                                                                         10/17
libsemanage.semanage_direct_install_info: Overriding pcp module at lower priority 100 with module at priority 200.

  Installing       : pcp-conf-6.0.1-4.0.2.el9.x86_64                                                                            11/17
  Installing       : pcp-libs-6.0.1-4.0.2.el9.x86_64                                                                            12/17
  Running scriptlet: pcp-6.0.1-4.0.2.el9.x86_64                                                                                 13/17
Creating group 'pcp' with GID 977.
Creating user 'pcp' (Performance Co-Pilot) with UID 977 and GID 977.

  Installing       : pcp-6.0.1-4.0.2.el9.x86_64                                                                                 13/17
  Running scriptlet: pcp-6.0.1-4.0.2.el9.x86_64                                                                                 13/17
  Installing       : python3-pcp-6.0.1-4.0.2.el9.x86_64                                                                         14/17
  Installing       : pcp-system-tools-6.0.1-4.0.2.el9.x86_64                                                                    15/17
  Installing       : pcp-gui-6.0.1-4.0.2.el9.x86_64                                                                             16/17
  Installing       : pcp-doc-6.0.1-4.0.2.el9.noarch                                                                             17/17
  Running scriptlet: pcp-doc-6.0.1-4.0.2.el9.noarch                                                                             17/17
  Verifying        : pcp-6.0.1-4.0.2.el9.x86_64                                                                                  1/17
  Verifying        : pcp-conf-6.0.1-4.0.2.el9.x86_64                                                                             2/17
```

```
  Verifying        : pcp-doc-6.0.1-4.0.2.el9.noarch                                                 3/17
  Verifying        : pcp-gui-6.0.1-4.0.2.el9.x86_64                                                 4/17
  Verifying        : pcp-libs-6.0.1-4.0.2.el9.x86_64                                                5/17
  Verifying        : pcp-selinux-6.0.1-4.0.2.el9.x86_64                                             6/17
  Verifying        : pcp-system-tools-6.0.1-4.0.2.el9.x86_64                                        7/17
  Verifying        : pcre2-utf16-10.40-2.0.2.el9.x86_64                                             8/17
  Verifying        : python3-pcp-6.0.1-4.0.2.el9.x86_64                                             9/17
  Verifying        : qt5-qtbase-5.15.3-1.el9.x86_64                                                10/17
  Verifying        : qt5-qtbase-common-5.15.3-1.el9.noarch                                         11/17
  Verifying        : qt5-qtbase-gui-5.15.3-1.el9.x86_64                                            12/17
  Verifying        : qt5-qtsvg-5.15.3-1.el9.x86_64                                                 13/17
  Verifying        : xcb-util-image-0.4.0-19.el9.x86_64                                            14/17
  Verifying        : xcb-util-keysyms-0.4.0-17.el9.x86_64                                          15/17
  Verifying        : xcb-util-renderutil-0.3.9-20.el9.x86_64                                       16/17
Verifying        : xcb-util-wm-0.4.1-22.el9.x86_64                                                17/17

Installed:
  pcp-6.0.1-4.0.2.el9.x86_64                    pcp-conf-6.0.1-4.0.2.el9.x86_64          pcp-doc-6.0.1-4.0.2.el9.noarch
  pcp-gui-6.0.1-4.0.2.el9.x86_64                pcp-libs-6.0.1-4.0.2.el9.x86_64          pcp-selinux-6.0.1-4.0.2.el9.x86_64
  pcp-system-tools-6.0.1-4.0.2.el9.x86_64       pcre2-utf16-10.40-2.0.2.el9.x86_64       python3-pcp-6.0.1-4.0.2.el9.x86_64
  qt5-qtbase-5.15.3-1.el9.x86_64                qt5-qtbase-common-5.15.3-1.el9.noarch    qt5-qtbase-gui-5.15.3-1.el9.x86_64
  qt5-qtsvg-5.15.3-1.el9.x86_64                 xcb-util-image-0.4.0-19.el9.x86_64       xcb-util-keysyms-0.4.0-17.el9.x86_64
  xcb-util-renderutil-0.3.9-20.el9.x86_64       xcb-util-wm-0.4.1-22.el9.x86_64

Complete!
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot Installation - Collector

Unlike the previous installation using the pcp-zeroconf RPM, the Performance Co-Pilot services will not be automatically started once this installation has completed.

Several configuration changes now need to be made prior to the enabling and starting of the PCP services.

Advanced PCP Configuration

The following changes now need to be made.

Enable PCP base services

Using the 'systemctl' command, enable the three base services to restart on reboot:

```
[root@brm-pcapm-pcp-02 ~]# systemctl enable pmcd pmlogger pmie
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot Enable Base Services - Collector

Network Interface change for pmcd

Append the following text '-i <system IP address>' to the '/etc/pcp/pmcd/pmcd.options' configuration file to enable the pmcd service to be addressable by an external collector system.

Firewall Changes

Configure the default firewall services to permit the pmcd daemon to be visible to the external collation service:

```
[root@brm-pcapm-pcp-02 ~]# firewall-cmd --add-service=pmcd --permanent
success
[root@brm-pcapm-pcp-02 ~]# firewall-cmd --reload
success
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot Enable PMCD Firewall Access - Collector

Configure for Remote PCP Archive Log Collection

By default, the PCP archive logger (pmlogger) service will only be archiving the performance metrics collected by the local performance metrics collection daemon (pmcd). A new file, name 'remote', needs to be created within the '/etc/pcp/pmlogger/control.d' directory to instruct the archive logger to collect performance metrics from remote VM Instances.

The command 'pmfind -s pmcd -m probe=<CIDR block>' will find all pcmd services within the specified networking subnet that are visible.

For example:

```
[root@brm-pcapm-pcp-02 ~]# pmfind -s pmcd -m probe=10.80.160.0/24
discovered pmcd servers:
  pcp://10.80.160.20:44321
  pcp://10.80.160.48:44321
  pcp://10.80.160.68:44321
  pcp://10.80.160.70:44321
  pcp://10.80.160.79:44321
  pcp://10.80.160.78:44321
  pcp://10.80.160.81:44321
  pcp://10.80.160.93:44321
  pcp://10.80.160.92:44321
  pcp://10.80.160.91:44321
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot – using the pmfind utility

The contents of the 'remote' should mirror the example used below:

```
#
# PCP archive logging configuration/control
#
# see ../control for a description of the format
#

# === VARIABLE ASSIGNMENTS ===
#
# DO NOT REMOVE OR EDIT THE FOLLOWING LINE
$version=1.1

# Uncomment one of the lines below to enable/disable compression behaviour
# that is different to the pmlogger_daily default.
# Value is days before compressing archives, 0 is immediate compression,
# "never" or "forever" suppresses compression.
#
#$PCP_COMPRESSAFTER=0
$PCP_COMPRESSAFTER=3
#$PCP_COMPRESSAFTER=never
```

```
# === LOGGER CONTROL SPECIFICATIONS ===
#
#Host                                P?      S?      directory                                   args
ca-ovsx54.us.oracle.com              n       n       PCP_LOG_DIR/pmlogger/ca-ovsx54              -r -T24h10m -c config.ca-ovsx54 -v 100Mb
ca-ovsx69.us.oracle.com              n       n       PCP_LOG_DIR/pmlogger/ca-ovsx69              -r -T24h10m -c config.ca-ovsx69 -v 100Mb
srd-pcp-test-01.us.oracle.com        n       n       PCP_LOG_DIR/pmlogger/srd-pcp-test-01        -r -T24h10m -c config.srd-pcp-test-01 -v 100Mb
srd-pcp-test-02.us.oracle.com        n       n       PCP_LOG_DIR/pmlogger/srd-pcp-test-02        -r -T24h10m -c config.srd-pcp-test-02 -v 100Mb
srd-pcp-test-03.us.oracle.com        n       n       PCP_LOG_DIR/pmlogger/srd-pcp-test-03        -r -T24h10m -c config.srd-pcp-test-03 -v 100Mb
srd-external-grafana.us.oracle.com   n       n       PCP_LOG_DIR/pmlogger/srd-external-grafana   -r -T24h10m -c config.srd-external-grafana -v 100Mb
brm-pcapm-mysqldb01.us.oracle.com    n       n       PCP_LOG_DIR/pmlogger/brm-pcapm-mysqldb01    -r -T24h10m -c config.brm-pcapm-mysqldb01 -v 100Mb
srd-vmware-ol6u10.us.oracle.com      n       n       PCP_LOG_DIR/pmlogger/srd-vmware-ol6u10      -r -T24h10m -c config.srd-vmware-ol6u10 -v 100Mb
srd-vmware-ol7u9.us.oracle.com       n       n       PCP_LOG_DIR/pmlogger/srd-vmware-ol7u9       -r -T24h10m -c config.srd-vmware-ol7u9 -v 100Mb
srd-vmware-ol8u5.us.oracle.com       n       n       PCP_LOG_DIR/pmlogger/srd-vmware-ol8u5       -r -T24h10m -c config.srd-vmware-ol8u5 -v 100Mb
10.80.160.55                         n       n       PCP_LOG_DIR/pmlogger/brm-pcapm-pcp-grafana  -r -T24h10m -c config.brm-pcapm-pcp-grafana -v 100Mb
10.80.160.68                         n       n       PCP_LOG_DIR/pmlogger/brm-pcapm-pcp-01       -r -T24h10m -c config.brm-pcapm-pcp-01 -v 100Mb
```

Performance Co-Pilot example pmlogger remote configuration file

In the example above, a mixture of physical host systems, Oracle Distributed Cloud-based VM Instances and even some VMWare-based Virtual Machines have been included within the remote host configuration file.

Start and check status for PCP services

The base PCP services can now be started, and their status' checked:

```
[root@brm-pcapm-pcp-02 ~]# systemctl start pmcd pmlogger pmie
[root@brm-pcapm-pcp-02 ~]# systemctl status pmcd pmlogger pmie
● pmcd.service - Performance Metrics Collector Daemon
     Loaded: loaded (/usr/lib/systemd/system/pmcd.service; enabled; preset: disabled)
     Active: active (running) since Sun 2023-07-30 06:44:49 BST; 2h 30min ago
       Docs: man:pmcd(1)
   Main PID: 241648 (pmcd)
      Tasks: 6 (limit: 98963)
     Memory: 9.3M
        CPU: 4.305s
     CGroup: /system.slice/pmcd.service
             ├─241648 /usr/libexec/pcp/bin/pmcd -i 10.80.160.48 -A
             ├─241652 /var/lib/pcp/pmdas/root/pmdaroot
             ├─241653 /var/lib/pcp/pmdas/proc/pmdaproc -d 3
             ├─241655 /var/lib/pcp/pmdas/xfs/pmdaxfs -d 11
             ├─241656 /var/lib/pcp/pmdas/linux/pmdalinux
             └─241657 /var/lib/pcp/pmdas/kvm/pmdakvm -d 95

Jul 30 06:44:48 brm-pcapm-pcp-02 systemd[1]: Starting Performance Metrics Collector Daemon...
Jul 30 06:44:49 brm-pcapm-pcp-02 systemd[1]: Started Performance Metrics Collector Daemon.

● pmlogger.service - Performance Metrics Archive Logger
     Loaded: loaded (/usr/lib/systemd/system/pmlogger.service; enabled; preset: disabled)
     Active: active (running) since Sun 2023-07-30 06:44:51 BST; 2h 30min ago
       Docs: man:pmlogger(1)
   Main PID: 242172 (pmlogger)
      Tasks: 1 (limit: 98963)
     Memory: 7.4M
        CPU: 1.330s
     CGroup: /system.slice/pmlogger.service
             └─242172 /usr/libexec/pcp/bin/pmlogger -N -P -r -T24h10m -c config.default -v 100mb -m pmlogger_check %Y%m%d.%H.%M

Jul 30 06:44:49 brm-pcapm-pcp-02 systemd[1]: Starting Performance Metrics Archive Logger...
Jul 30 06:44:51 brm-pcapm-pcp-02 systemd[1]: Started Performance Metrics Archive Logger.

● pmie.service - Performance Metrics Inference Engine
     Loaded: loaded (/usr/lib/systemd/system/pmie.service; enabled; preset: disabled)
     Active: active (running) since Sun 2023-07-30 06:44:50 BST; 2h 30min ago
       Docs: man:pmie(1)
   Main PID: 242162 (pmie)
      Tasks: 1 (limit: 98963)
     Memory: 1.4M
        CPU: 624ms
     CGroup: /system.slice/pmie.service
             └─242162 /usr/bin/pmie -b -F -P -l /var/log/pcp/pmie/brm-pcapm-pcp-02/pmie.log -c config.default

Jul 30 06:44:49 brm-pcapm-pcp-02 systemd[1]: Starting Performance Metrics Inference Engine...
Jul 30 06:44:50 brm-pcapm-pcp-02 systemd[1]: Started Performance Metrics Inference Engine.
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot Basic Services Status - Collector

All services are Active and Enabled.

Check the PCP Remote Archive Logging activity

Now that the PCP Archive Logger service (pmlogger) is running, after a few minutes check the contents of the default archive log directory ('/var/log/pcp/pmlogger').

For the system used in the example, the output looks as follows:

```
[root@brm-pcapm-pcp-02 pmlogger]# ls -lrt
total 56
-rw-r--r--. 1 pcp pcp   33 Jul 30 00:10 pmlogger_daily.stamp
-rw-r--r--. 1 pcp pcp  151 Jul 30 06:45 pmlogger_check.log.prev
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 brm-pcapm-pcp-02
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 ca-ovsx54
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 ca-ovsx69
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 srd-pcp-test-01
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 srd-pcp-test-02
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 srd-pcp-test-03
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 srd-external-grafana
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 brm-pcapm-mysqldb01
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 srd-vmware-ol6u10
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 srd-vmware-ol7u9
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 srd-vmware-ol8u5
drwxr-xr-x. 2 pcp pcp  119 Jul 30 08:55 brm-pcapm-pcp-grafana
drwxr-xr-x. 2 pcp pcp 4096 Jul 30 08:55 brm-pcapm-pcp-01
[root@brm-pcapm-pcp-02 pmlogger]# pwd
/var/log/pcp/pmlogger
[root@brm-pcapm-pcp-02 pmlogger]#
```

Performance Co-Pilot – Collector remote PCP Archive Logs

The local (collector) VM Instance, brm-pcapm-pcp-02, and the twelve remote PCP pmcd services within the remote pmlogger configuration as shown as being both present and active.

Making the collated PCP performance metrics available for Grafana

With the VM Instance collecting and collating the PCP performance metrics from each individual VM Instance's Performance Metrics Collection Daemon (pmcd) and successfully maintaining PCP Archive Logs for each configured host system, several further services need to be installed, configured, and enabled to provide the necessary connectivity to a Grafana Server service.

Install Redis

Redis, or **Re**mote **Di**ctionary **S**erver (redis), is an open source (BSD licensed), in-memory data structure store used as a database, cache, message broker, and streaming engine. Redis provides data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes, and streams. Redis has built-in replication, Lua scripting, LRU eviction, transactions, and different levels of on-disk persistence.

This provides the common 'data store' used by Grafana to access the Performance Co-Pilot performance metrics.

First the redis RPM needs to be installed on the Collector VM Instance:

```
[root@brm-pcapm-pcp-02 ~]# dnf install redis
Last metadata expiration check: 3:09:44 ago on Sat 29 Jul 2023 09:06:13 BST.
Dependencies resolved.
================================================================================================================================================
 Package                        Architecture          Version                       Repository                                    Size
================================================================================================================================================
Installing:
 redis                          x86_64                6.2.7-1.el9                   ol9_appstream                                1.3 M

Transaction Summary
================================================================================================================================================
Install  1 Package

Total download size: 1.3 M
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
redis-6.2.7-1.el9.x86_64.rpm                                                                                  2.5 MB/s | 1.3 MB     00:00
------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                                         2.5 MB/s | 1.3 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                                      1/1
  Running scriptlet: redis-6.2.7-1.el9.x86_64                                                                                             1/1
  Installing       : redis-6.2.7-1.el9.x86_64                                                                                             1/1
  Running scriptlet: redis-6.2.7-1.el9.x86_64                                                                                             1/1
  Verifying        : redis-6.2.7-1.el9.x86_64                                                                                             1/1

Installed:
  redis-6.2.7-1.el9.x86_64

Complete!
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot – Collector install redis

Enable and start the redis data caching service

Once installed, the redis service needs to be both enabled and started:

```
[root@brm-pcapm-pcp-02 ~]# systemctl enable redis
[root@brm-pcapm-pcp-02 ~]# systemctl start redis
[root@brm-pcapm-pcp-02 ~]# systemctl status redis
● redis.service - Redis persistent key-value database
     Loaded: loaded (/usr/lib/systemd/system/redis.service; enabled; preset: disabled)
    Drop-In: /etc/systemd/system/redis.service.d
             └─limit.conf
     Active: active (running) since Sat 2023-07-29 11:20:50 BST; 22h ago
   Main PID: 51294 (redis-server)
     Status: "Ready to accept connections"
      Tasks: 5 (limit: 98963)
     Memory: 753.9M
        CPU: 42min 16.425s
     CGroup: /system.slice/redis.service
             └─51294 "/usr/bin/redis-server 127.0.0.1:6379"

Jul 29 11:20:49 brm-pcapm-pcp-02 systemd[1]: Starting Redis persistent key-value database...
Jul 29 11:20:50 brm-pcapm-pcp-02 systemd[1]: Started Redis persistent key-value database.
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot – Collector redis status check

The redis service is now ready for use.

Enable and start the PCP Proxy service (pmproxy)

The PCP pmproxy service makes use of the redis data store to expose the collected PCP Archive Logs externally.

This PCP component was installed with the base pcp RPM package, but does need to be enabled and started:

```
[root@brm-pcapm-pcp-02 ~]# systemctl enable pmproxy
[root@brm-pcapm-pcp-02 ~]# systemctl start pmproxy
[root@brm-pcapm-pcp-02 ~]# systemctl status pmproxy
• pmproxy.service - Proxy for Performance Metrics Collector Daemon
     Loaded: loaded (/usr/lib/systemd/system/pmproxy.service; enabled; preset: disabled)
     Active: active (running) since Sun 2023-07-30 06:44:49 BST; 3h 12min ago
       Docs: man:pmproxy(1)
   Main PID: 241661 (pmproxy)
      Tasks: 1 (limit: 98963)
     Memory: 213.3M
        CPU: 47.760s
     CGroup: /system.slice/pmproxy.service
             └─241661 /usr/libexec/pcp/bin/pmproxy -F -A

Jul 30 06:44:49 brm-pcapm-pcp-02 systemd[1]: Starting Proxy for Performance Metrics Collector Daemon...
Jul 30 06:44:49 brm-pcapm-pcp-02 systemd[1]: Started Proxy for Performance Metrics Collector Daemon.
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot – Collector pmproxy status check

The pmproxy service is now ready for use.

Check local caching is occurring

With both additional services installed and operational, a few simple checks will make sure that the pmproxy service is accessing redis.

Three simple checks will be performed:

- run the command 'pmseries disk.dev.read' – this will display a list of the data series being cached as hashed values
  - adding '| wc -l' command will provide a count of the data series being collected – in this step-by-step guide a value of 13 is expected
- run the command 'redis-cli dbsize' – this will display the current (in memory) size of the redis cache
- run the command 'cat /var/log/pcp/pmproxy/pmproxy.log | grep Info' – this will display the information status from the pmproxy
  - Any errors reported will require further investigation

The following code block shows the responses to all the above commands from the example system being used:

```
[root@brm-pcapm-pcp-02 ~]# pmseries disk.dev.read
0df4db1c6577f5438088e4e6fab68596fe3641bc
17b44656a262e9d258b00834d35254d72ea500a9
1a18e9ce68012ef9940e5eedb100405f16b8159d
5a170d2911af2da200cc5ee0d75317ec59969e65
5b60963e4459e746b95fbbbaa50880dd317466e0
5d99e0db88ee82328ace1e4bfdfe045fdae6d40d
61d3c63a709db58b34c7c42b7b7e8a24eda6ae90
661a81c4efbad8e572234461c9595f40546a1985
7d361040220f09040686f0838aa11fad5bdbb859
7e039f16e2340533d5a38d4a2613e86ae8c191d6
8c6a4af025fbf9cefb4f2f734a2be5a446880c4a
c6655189dfcaccd6001a0a28e06c40fa2a8ea11b
d9be381573496b0c6276ff325262fe2a7cc239fa
[root@brm-pcapm-pcp-02 ~]#
[root@brm-pcapm-pcp-02 ~]# pmseries disk.dev.read | wc -l
13
[root@brm-pcapm-pcp-02 ~]#

[root@brm-pcapm-pcp-02 ~]# redis-cli dbsize
(integer) 62252

[root@brm-pcapm-pcp-02 ~]# cat /var/log/pcp/pmproxy/pmproxy.log | grep Info
[Sun Jul 30 06:44:49] pmproxy(241661) Info: Redis slots, command keys, schema version setup
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot – Collector redis & pmproxy redis checks

All is as expected.

External Access for PCP Proxy (pmproxy)

The final configuration change is to permit external connectivity to the pmproxy service from external systems. To complete this action, a simple firewall change and reload is required:

```
[root@brm-pcapm-pcp-02 ~]#  firewall-cmd --add-service=pmproxy --permanent
success
[root@brm-pcapm-pcp-02 ~]# firewall-cmd --reload
success
[root@brm-pcapm-pcp-02 ~]# firewall-cmd --list-all
public (active)
  target: default
  icmp-block-inversion: no
  interfaces: ens33
  sources:
  services: cockpit dhcpv6-client pmcd pmproxy ssh
  ports:
  protocols:
  forward: yes
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
[root@brm-pcapm-pcp-02 ~]#
```

Performance Co-Pilot – Collector pmproxy firewall changes

The required firewall changes are now in place.

Next Steps

The Performance Co-Pilot metrics collection and collation VM Instance described within this section of the document provides a single source for further analysis of the collected metrics.

The Performance Co-Pilot framework provide numerous tools that can be utilized at the command line for the detailed analysis of the collated data sets.

PCP provide their own excellent documentation on the analysis capabilities for each of these tools. Please see the following documentation for further details:

- PCP Monitoring Systems Performance – https://pcp.readthedocs.io/en/latest/UAG/MonitoringSystemPerformance.html
- PCP Performance Metrics Inference Engine (pmie) – https://pcp.readthedocs.io/en/latest/UAG/PerformanceMetricsInferenceEngine.html
- PCP Using Archive Logs with Performance Tools – https://pcp.readthedocs.io/en/latest/UAG/ArchiveLogging.html#using-archive-logs-with-performance-tools
- PCP Deployment Strategies – https://pcp.readthedocs.io/en/latest/UAG/PcpDeploymentStrategies.html

Finally, consideration should be given any Visualization requirements for metrics Observability, Monitoring & Alerting requirements. The use of Grafana as a Visualization tool is strongly encouraged.

For customer deployments that make use of only a single Oracle Distributed Cloud domain, then the Grafana server service could co-exist alongside the collection and collation of PCP metrics inside a single VM Instance.

Where multiple Oracle Distributed Cloud domains are present, then the use of an external Grafana server service is strongly encouraged.

The final section of this document will outline how to configure Grafana to access the PCP metrics being collected.

This completes this section of the step-by-step guide.

Section References

The following URL's provide links to additional documentation:

- Performance Co-Pilot documentation – https://pcp.readthedocs.io/en/latest/
- Performance Co-Pilot documentation – Quick Reference Guide – https://pcp.readthedocs.io/en/latest/OG/QuickReferenceGuide.html
- Performance Co-Pilot documentation – PCP Archive Logger Deployment – https://pcp.readthedocs.io/en/latest/UAG/PcpDeploymentStrategies.html#pcp-archive-logger-deployment
- Performance Co-Pilot documentation - PCP Redis – https://grafana-pcp.readthedocs.io/en/latest/datasources/redis.html
- Performance Co-Pilot documentation – Troubleshooting – https://grafana-pcp.readthedocs.io/en/latest/troubleshooting.html
- Redis Documentation – https://redis.io/docs/about/

Adding Performance Co-Pilot to a Grafana Server Service

This document assumes that the Grafana Server service has already been installed and will concentrate on the following:

- Installation of the Grafana – Performance Co-Pilot RPM package
- Installation of the Performance Co-Pilot plugin
- Enabling the Performance Co-Pilot plugin
- Creating a new data source
- Accessing the Grafana Dashboard(s) provided within the Performance Co-Pilot plugin

Each step is now covered in detail below.

Installation – Grafana-pcp RPM package

In preparation for enabling the Grafana Server service to access the collected and collated PCP performance metrics, a single RPM package () needs to be installed. This package adds the required additional framework needed by the Grafana – Performance Co-Pilot plugin and associated components.

A simple RPM package is installed as follows:

```
[root@brm-pcapm-pcp-grafana ~]# dnf install grafana-pcp
Last metadata expiration check: 0:58:01 ago on Sun 30 Jul 2023 05:22:40 BST.
Dependencies resolved.
================================================================================================================================================
 Package                         Architecture         Version                    Repository                                  Size
================================================================================================================================================
Installing:
 grafana-pcp                     x86_64               5.1.1-1.el9                ol9_appstream                               10 M

Transaction Summary
================================================================================================================================================
Install  1 Package

Total download size: 10 M
Installed size: 34 M
Is this ok [y/N]: y
Downloading Packages:
grafana-pcp-5.1.1-1.el9.x86_64.rpm                                                                          9.6 MB/s |  10 MB     00:01
------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                                       9.5 MB/s |  10 MB     00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                                                                      1/1
  Installing       : grafana-pcp-5.1.1-1.el9.x86_64                                                                                       1/1
  Running scriptlet: grafana-pcp-5.1.1-1.el9.x86_64                                                                                       1/1
Detected unsafe path transition /var/lib/grafana (owned by grafana) → /var/lib/grafana/plugins (owned by root) during canonicalization of
/var/lib/grafana/plugins.

Detected unsafe path transition /var/lib/grafana (owned by grafana) → /var/lib/grafana/plugins (owned by root) during canonicalization of
/var/lib/grafana/plugins.

  Verifying        : grafana-pcp-5.1.1-1.el9.x86_64                                                                                       1/1

Installed:
  grafana-pcp-5.1.1-1.el9.x86_64

Complete!
[root@brm-pcapm-pcp-grafana ~]#
```

Grafana Server – RPM package installation

In this case, two warnings have been displayed concerning ownership of the ‘plugins’ directory within the ‘/var/lib/grafana’ filesystem. A simple ‘chown -R grafana:grafana plugins’ command from within the ‘/var/lib/grafana’ filesystem fixes this issue.

```
[root@brm-pcapm-pcp-grafana ~]# cd /var/lib/grafana
[root@brm-pcapm-pcp-grafana grafana]# ls -l
total 1112
drwxr-x---. 3 grafana grafana      15 Jul 30 05:54 alerting
drwx------. 2 grafana grafana       6 Jul 30 05:54 csv
-rw-r-----. 1 grafana grafana 1134592 Jul 31 05:46 grafana.db
drwxr-xr-x. 3 root    root         40 Jul 30 06:24 plugins
drwx------. 2 grafana grafana       6 Jul 30 05:54 png
[root@brm-pcapm-pcp-grafana grafana]# chown -R grafana:grafana plugins
[root@brm-pcapm-pcp-grafana grafana]# ls -l
total 1112
drwxr-x---. 3 grafana grafana      15 Jul 30 05:54 alerting
drwx------. 2 grafana grafana       6 Jul 30 05:54 csv
-rw-r-----. 1 grafana grafana 1134592 Jul 31 05:46 grafana.db
drwxr-xr-x. 3 grafana grafana      40 Jul 30 06:24 plugins
drwx------. 2 grafana grafana       6 Jul 30 05:54 png
[root@brm-pcapm-pcp-grafana grafana]#
```

Grafana Server – /var/lib/Grafana/plugins ownership correction

This completes this step.

Installation – Performance Co-Pilot Grafana plugin

The performancecopilot-pcp-app now needs to be installed as a Grafana plugin. Using the 'grafana-cli' tool:

- List the available plugin for the installed version of Grafana Server – ('grafana-cli plugins list-remote | grep performance')
- Install the PCP performance app plugin – ('grafana-cli plugins install performancecopilot-pcp-app')

The expected output is shown below:

```
[root@brm-pcapm-pcp-grafana ~]# grafana-cli plugins list-remote | grep performance
id: performancecopilot-pcp-app version: 5.0.0
[root@brm-pcapm-pcp-grafana ~]# grafana cli plugins install performancecopilot-pcp-app
✓ Downloaded and extracted performancecopilot-pcp-app v5.0.0 zip successfully to /var/lib/grafana/plugins/performancecopilot-pcp-app

Please restart Grafana after installing or removing plugins. Refer to Grafana documentation for instructions if necessary.

[root@brm-pcapm-pcp-grafana ~]# cd /var/lib/grafana/
[root@brm-pcapm-pcp-grafana grafana]# ls -l
total 912
drwxr-x---. 3 grafana grafana     15 Jul 30 05:54 alerting
drwx------. 2 grafana grafana      6 Jul 30 05:54 csv
-rw-r-----. 1 grafana grafana 929792 Jul 30 06:20 grafana.db
drwxr-xr-x. 3 grafana grafana     40 Jul 30 06:24 plugins
drwx------. 2 grafana grafana      6 Jul 30 05:54 png
[root@brm-pcapm-pcp-grafana grafana]# cd plugins/
[root@brm-pcapm-pcp-grafana plugins]# ls -l
total 4
drwxr-xr-x. 7 grafana grafana    4096 Jul 30 06:24 performancecopilot-pcp-app
[root@brm-pcapm-pcp-grafana plugins]# cd performancecopilot-pcp-app/
[root@brm-pcapm-pcp-grafana performancecopilot-pcp-app]# ls -l
total 13156
-rw-r--r--. 1 grafana grafana   14963 Jul 30 06:24 CHANGELOG.md
drwxr-xr-x. 4 grafana grafana      33 Jul 30 06:24 dashboards
drwxr-xr-x. 5 grafana grafana      49 Jul 30 06:24 datasources
drwxr-xr-x. 2 grafana grafana      25 Jul 30 06:24 fonts
drwxr-xr-x. 3 grafana grafana      45 Jul 30 06:24 img
-rw-r--r--. 1 grafana grafana   11358 Jul 30 06:24 LICENSE
-rw-r--r--. 1 grafana grafana   11740 Jul 30 06:24 MANIFEST.txt
-rw-r--r--. 1 grafana grafana  245791 Jul 30 06:24 module.js
-rw-r--r--. 1 grafana grafana     149 Jul 30 06:24 module.js.LICENSE.txt
-rw-r--r--. 1 grafana grafana  820161 Jul 30 06:24 module.js.map
-rw-r--r--. 1 grafana grafana    3380 Jul 30 06:24 monaco-editor.js
-rw-r--r--. 1 grafana grafana   10741 Jul 30 06:24 monaco-editor.js.map
-rw-r--r--. 1 grafana grafana  163698 Jul 30 06:24 monaco-editor.worker.js
-rw-r--r--. 1 grafana grafana  682854 Jul 30 06:24 monaco-editor.worker.js.map
drwxr-xr-x. 5 grafana grafana      66 Jul 30 06:24 panels
-rw-r--r--. 1 grafana grafana    4820 Jul 30 06:24 plugin.json
-rw-r--r--. 1 grafana grafana    3509 Jul 30 06:24 README.md
-rw-r--r--. 1 grafana grafana 2530738 Jul 30 06:24 vendors~monaco-editor.js
-rw-r--r--. 1 grafana grafana     413 Jul 30 06:24 vendors~monaco-editor.js.LICENSE.txt
-rw-r--r--. 1 grafana grafana 8938803 Jul 30 06:24 vendors~monaco-editor.js.map
[root@brm-pcapm-pcp-grafana performancecopilot-pcp-app]# cd $HOME
[root@brm-pcapm-pcp-grafana ~]#
```

Grafana Server – PCP App plugin installation & verification

All is as expected.

Restart Grafana Server

As noted within the plugin installation, the Grafana Server service now needs to be restarted to pick up the newly installed PCP plugin:

```
[root@brm-pcapm-pcp-grafana ~]# systemctl restart grafana-server.service
[root@brm-pcapm-pcp-grafana ~]# systemctl status grafana-server.service
● grafana-server.service - Grafana instance
     Loaded: loaded (/usr/lib/systemd/system/grafana-server.service; enabled; preset: disabled)
     Active: active (running) since Sun 2023-07-30 06:26:24 BST; 8s ago
       Docs: http://docs.grafana.org
   Main PID: 7805 (grafana)
      Tasks: 15 (limit: 98963)
     Memory: 77.8M
        CPU: 3.123s
     CGroup: /system.slice/grafana-server.service
             ├─7805 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/var/run/grafana/grafana-server.pid
--packaging=rpm cfg:default.paths.logs=/var/log/grafana cfg:default.p>
             └─7812 /var/lib/grafana/plugins/performancecopilot-pcp-app/datasources/redis/pcp_redis_datasource_linux_amd64

Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=provisioning.alerting t=2023-07-30T06:26:24.236812263+01:00
level=info msg="finished to provision alerting"
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=modules t=2023-07-30T06:26:24.236994819+01:00 level=warn
msg="No modules registered..."
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=http.server t=2023-07-30T06:26:24.239136494+01:00 level=info
msg="HTTP Server Listen" address=[::]:3000 protocol=http subUrl= socket=
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=ngalert.state.manager t=2023-07-30T06:26:24.239198403+01:00
level=info msg="Warming state cache for startup"
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=ngalert.state.manager t=2023-07-30T06:26:24.239334831+01:00
level=info msg="State cache has been initialized" states=0 duration=135.7>
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=ticker t=2023-07-30T06:26:24.239405213+01:00 level=info
msg=starting first_tick=2023-07-30T06:26:30+01:00
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com systemd[1]: Started Grafana instance.
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=ngalert.multiorg.alertmanager t=2023-07-
30T06:26:24.240342834+01:00 level=info msg="starting MultiOrg Alertmanager"
Jul 30 06:26:24 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=grafanaStorageLogger t=2023-07-30T06:26:24.242026564+01:00
level=info msg="storage starting"
Jul 30 06:26:25 brm-pcapm-pcp-grafana.us.oracle.com grafana[7805]: logger=context userId=1 orgId=1 uname=admin t=2023-07-
30T06:26:25.061432213+01:00 level=info msg="Request Completed" method=GET path=/api/li>
[root@brm-pcapm-pcp-grafana ~]#
```

Grafana Server – restart Grafana Server services & check status

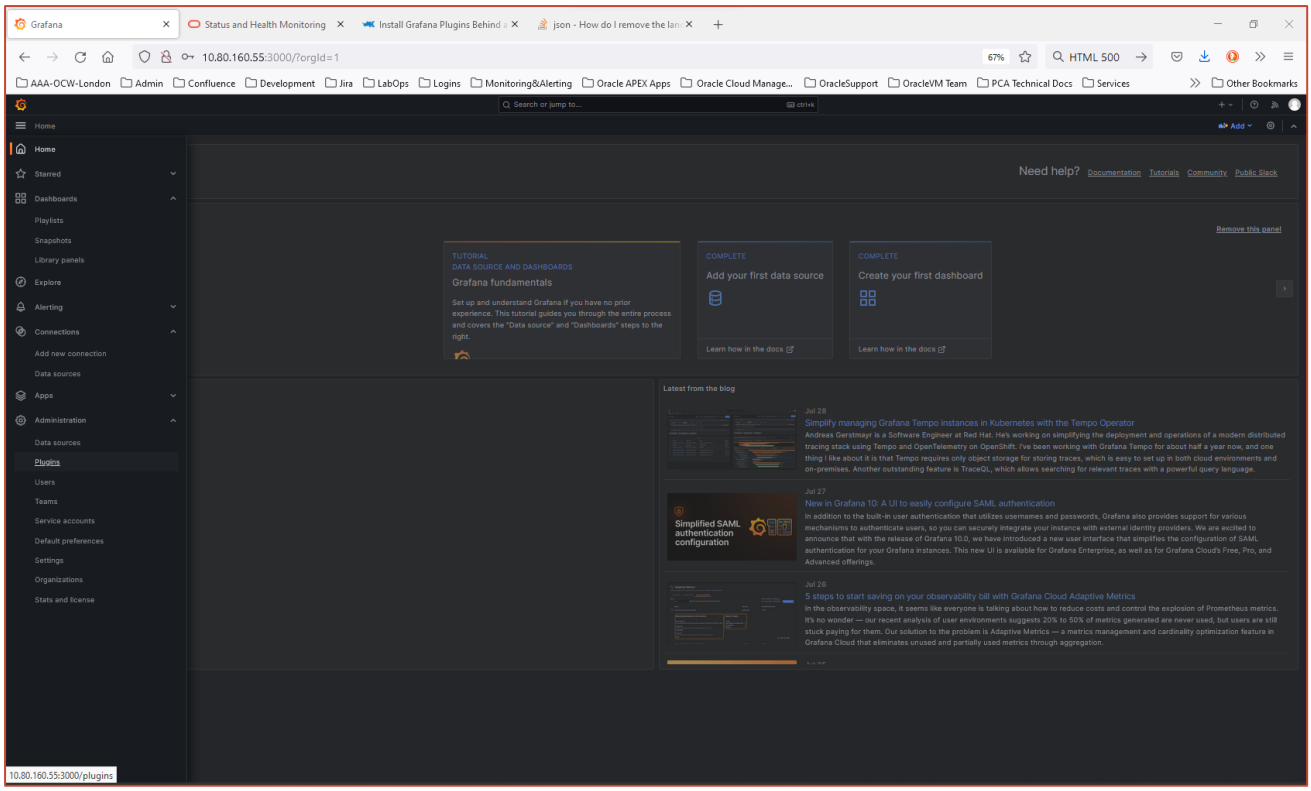
Note that the '/var/lib/grafana/plugins/performancecopilot-pcp-app' plugin is now loaded and running.

All is as expected.

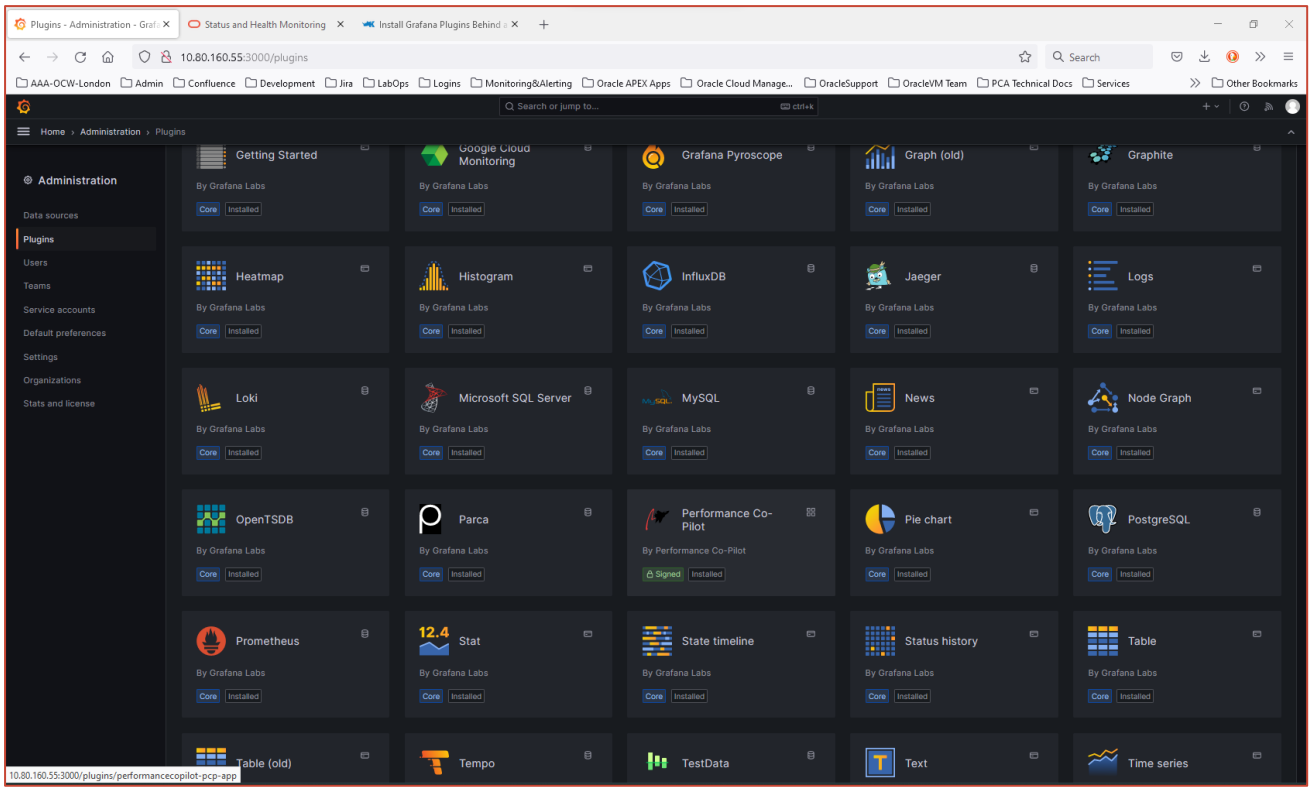
Enabling the – Performance Co-Pilot Grafana plugin

To complete the remaining activities, access now switches to the Grafana Server instance itself.

Access the Grafana Home Page and access the Plugin screens through the Administration menu:

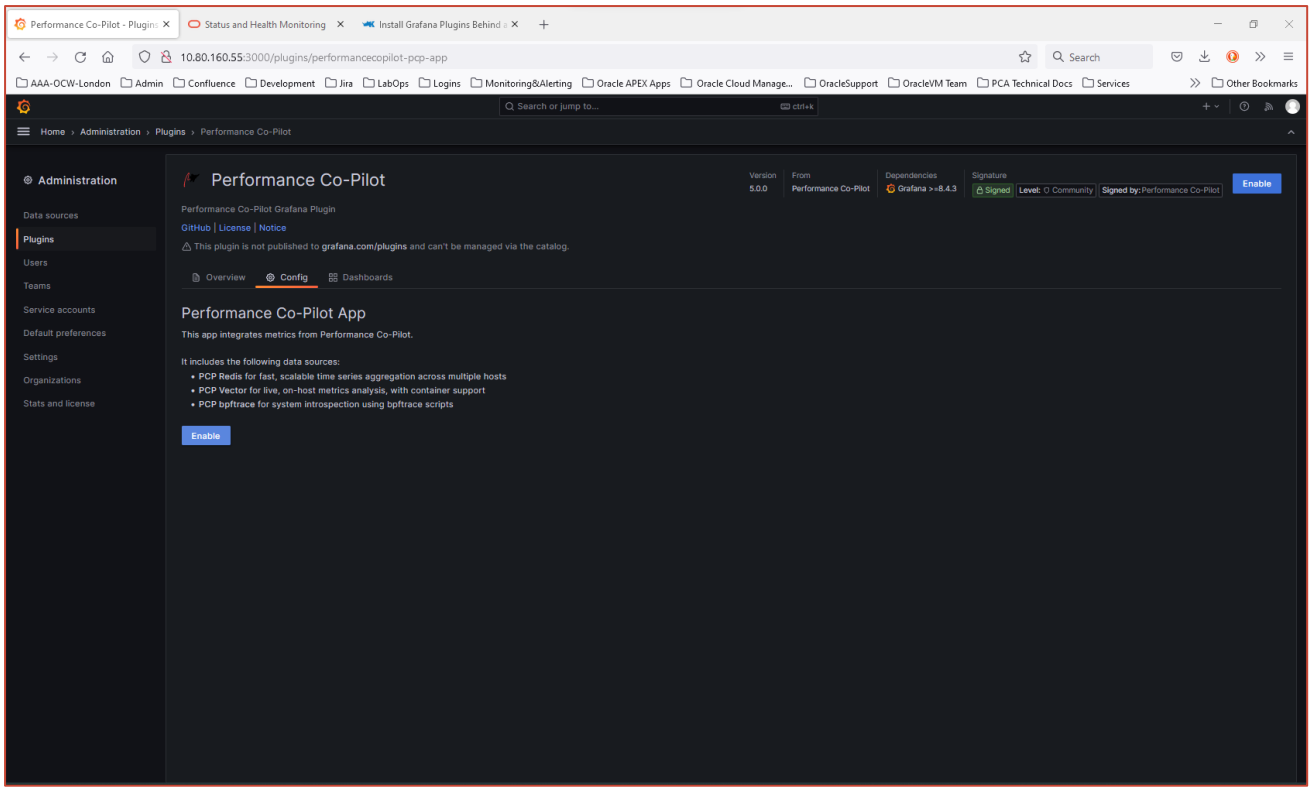


Grafana – Performance Co-Pilot Plugin – Home Page



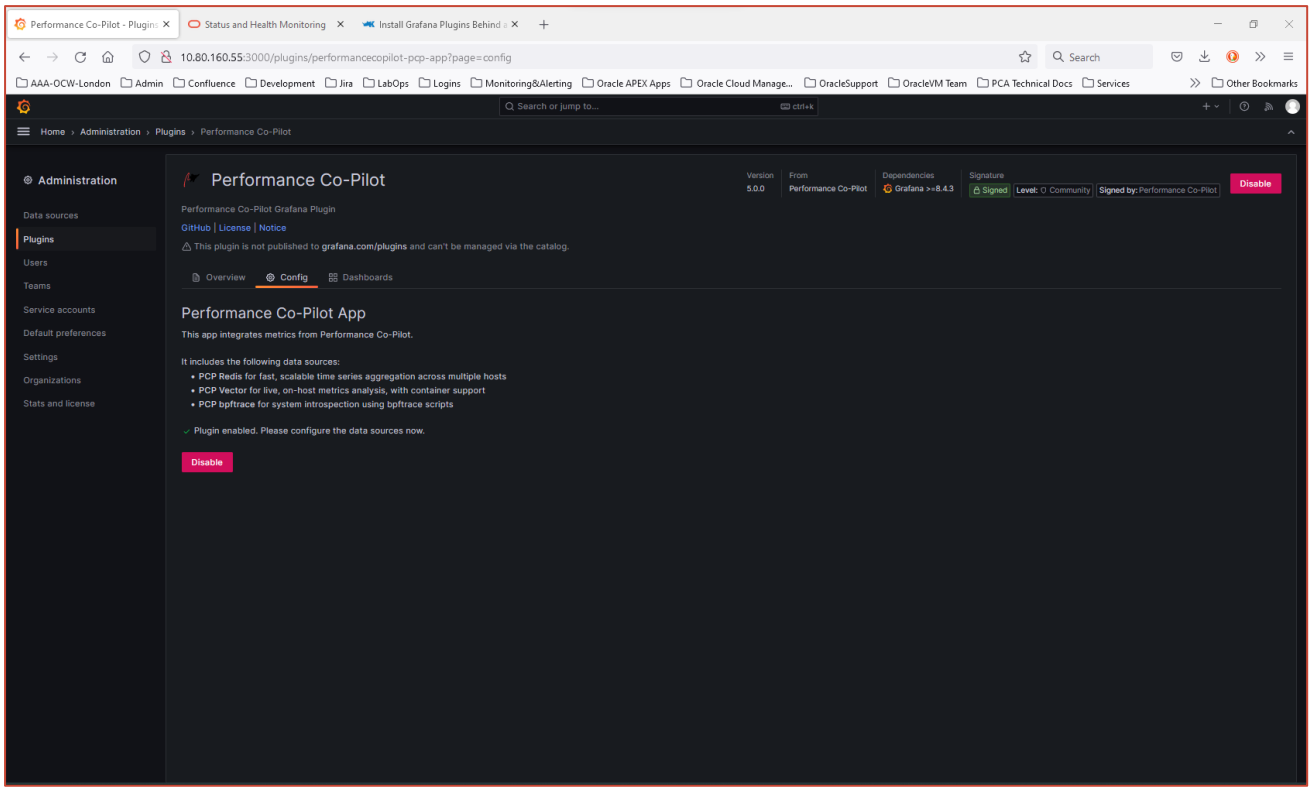
Grafana – Performance Co-Pilot Plugin - Select

Select the Performance Co-Pilot Plugin, which is ‘Installed’, but not yet enabled:



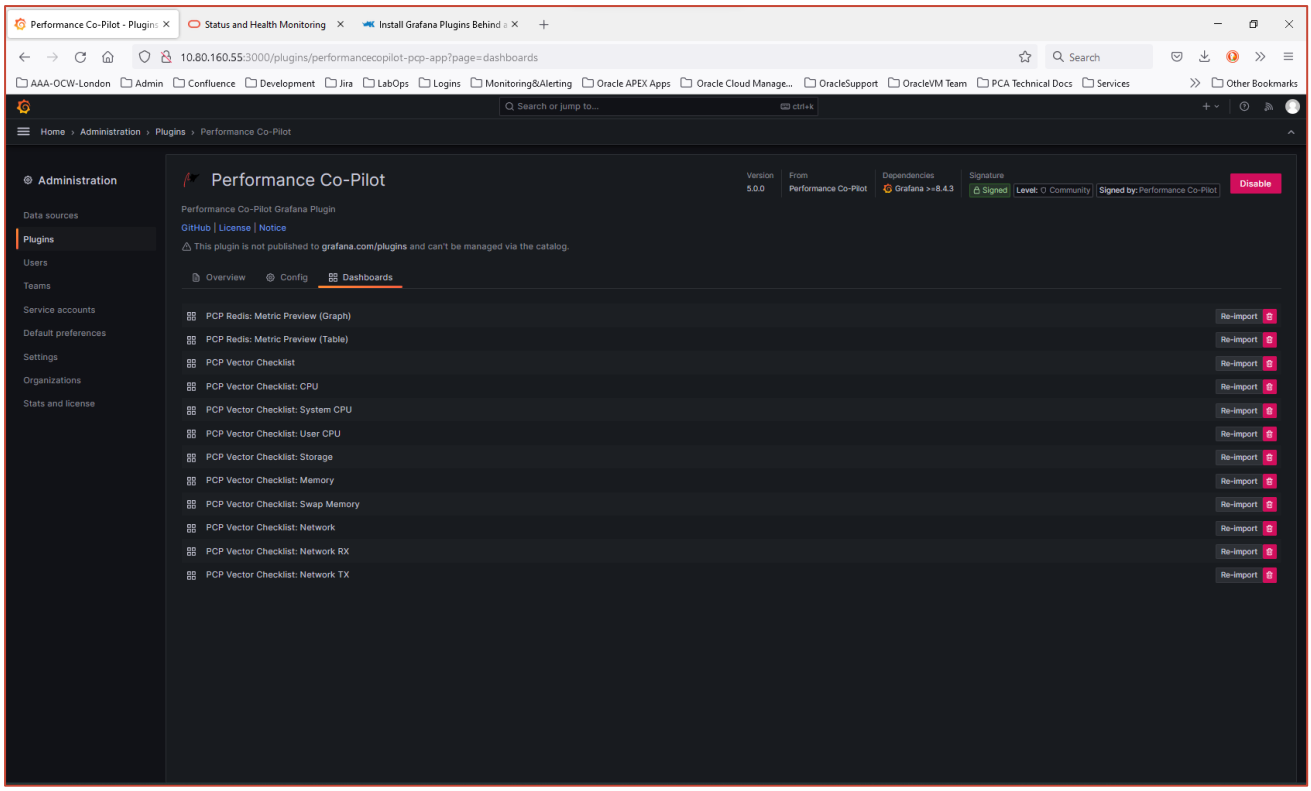
Grafana – Performance Co-Pilot Plugin - Enable

Enable the selected Grafana Plugin.



Grafana – Performance Co-Pilot Plugin - Enabled

Once enabled, check the ‘Dashboard’ tab to see the available Grafana Dashboards included with this Plugin App:



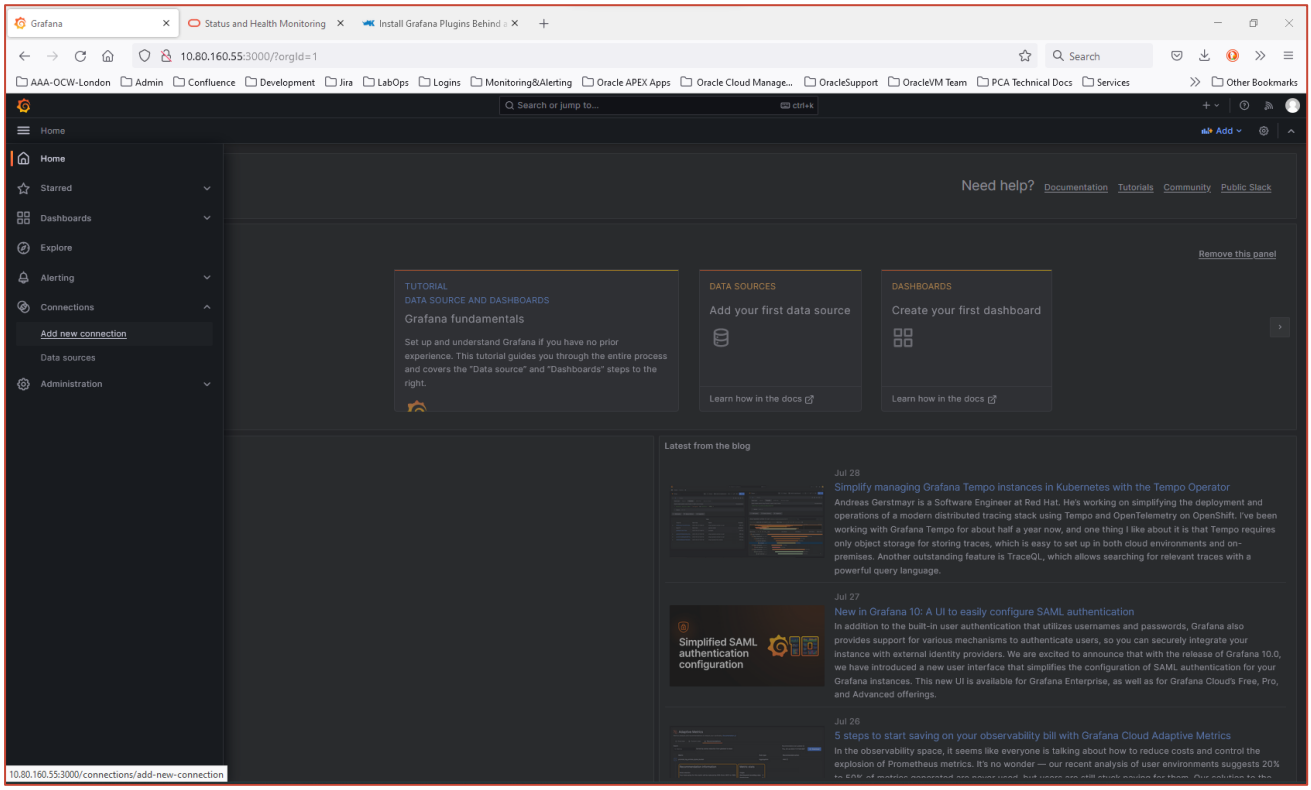
Grafana – Performance Co-Pilot Plugin – Included Dashboards

By default, these Grafana Dashboards will have been created within the 'General' Dashboard folder.

Creating the PCP Redis Data Source / Connection

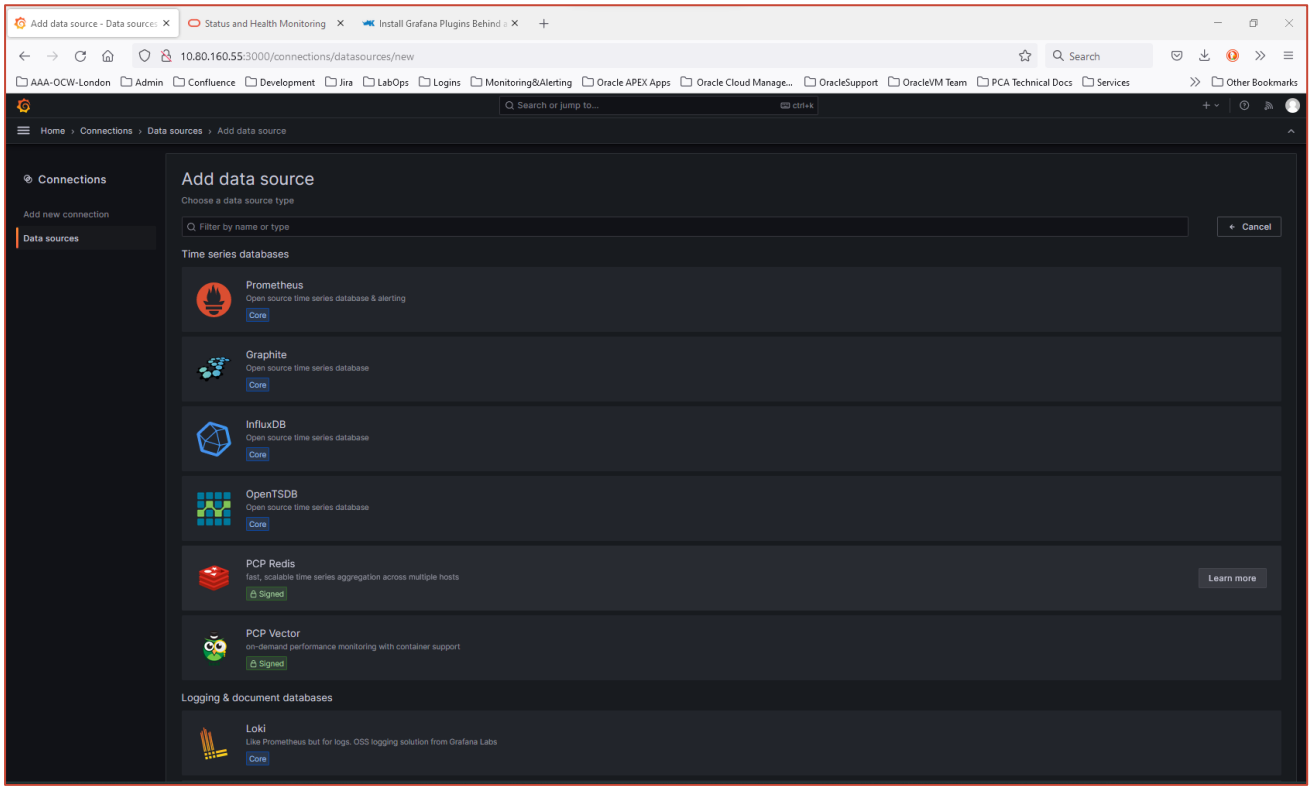
Now that the Performance Co-Pilot plugin (and associated application) have been enabled. A new Grafana Daat Source can be created.

Add a new connection through the Grafana Home Page Menu Bar:



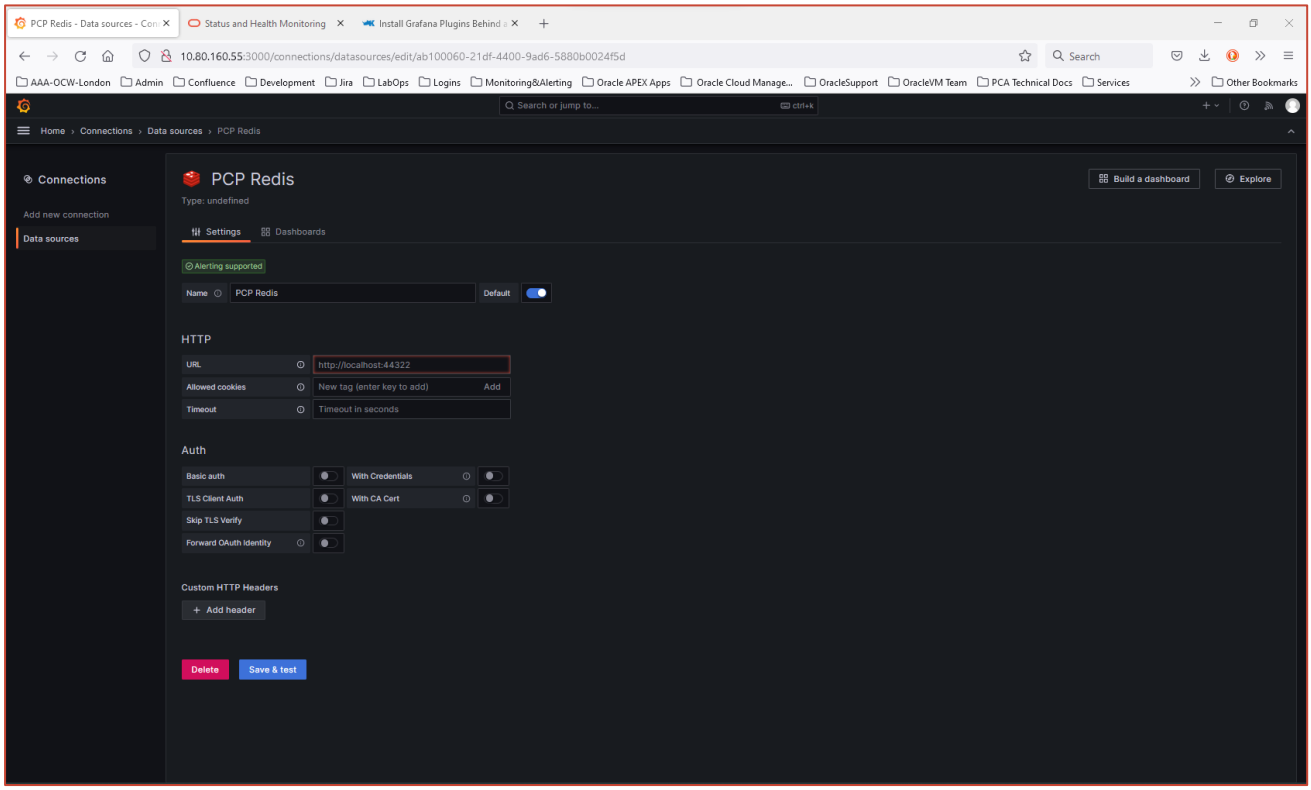
Grafana – New Data Source / Connection

Add the new Data Source as type 'PCP Redis':



Grafana – Add PCP Redis Data Source / Connection

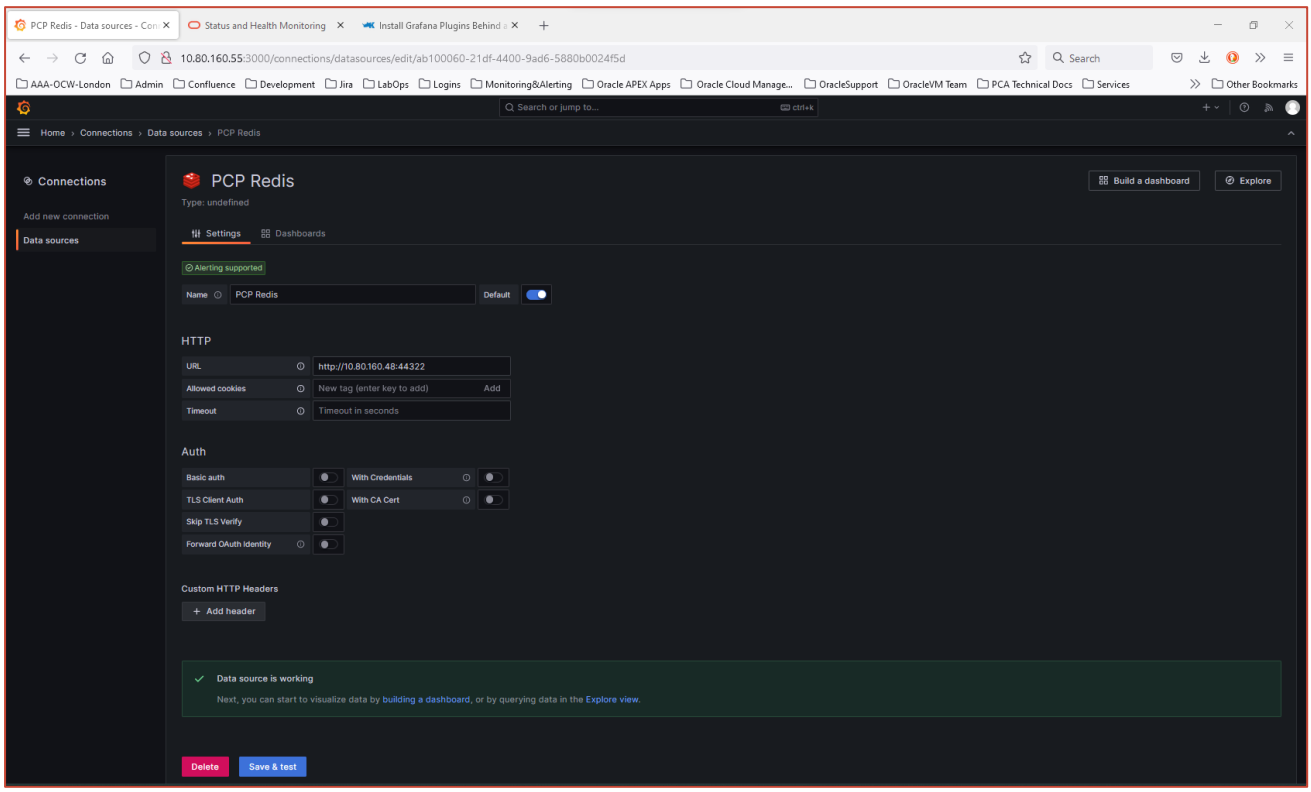
The screenshot below shows the Data Source settings required:



Grafana – Add PCP Redis Data Source / Connection – empty configuration

The PCP Redis Data Source just requires a name ('PCP Redis') and the URL for the VM Instance collector.

In this example a simple 'http://<IP address>:44322' format has been used. The use of a Fully Qualified Domain Name (FQDN) can be used if the relevant DNS records are in place:



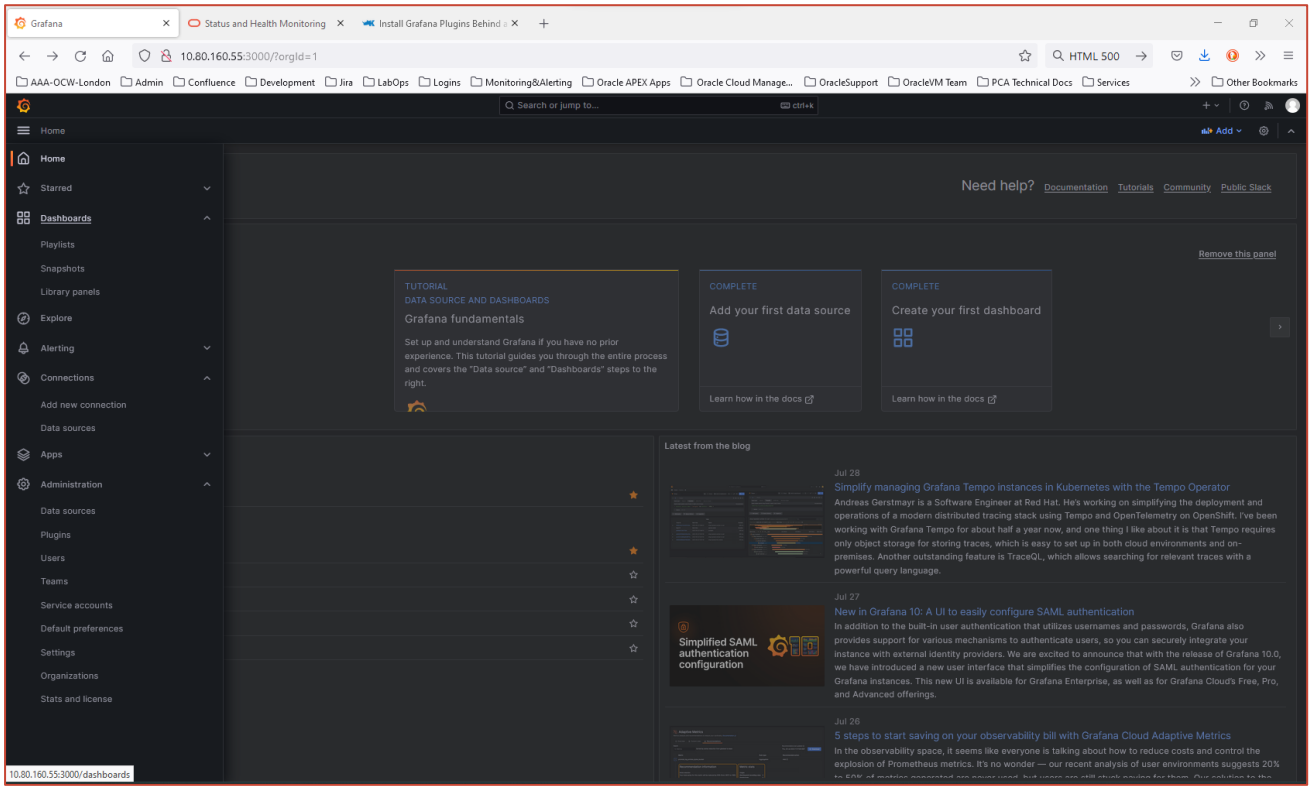
Grafana – Add PCP Redis Data Source / Connection – configured & tested

The above configuration has been saved and the new Data Source confirmed as ‘Working’.

Accessing the PCP Redis Grafana Dashboards

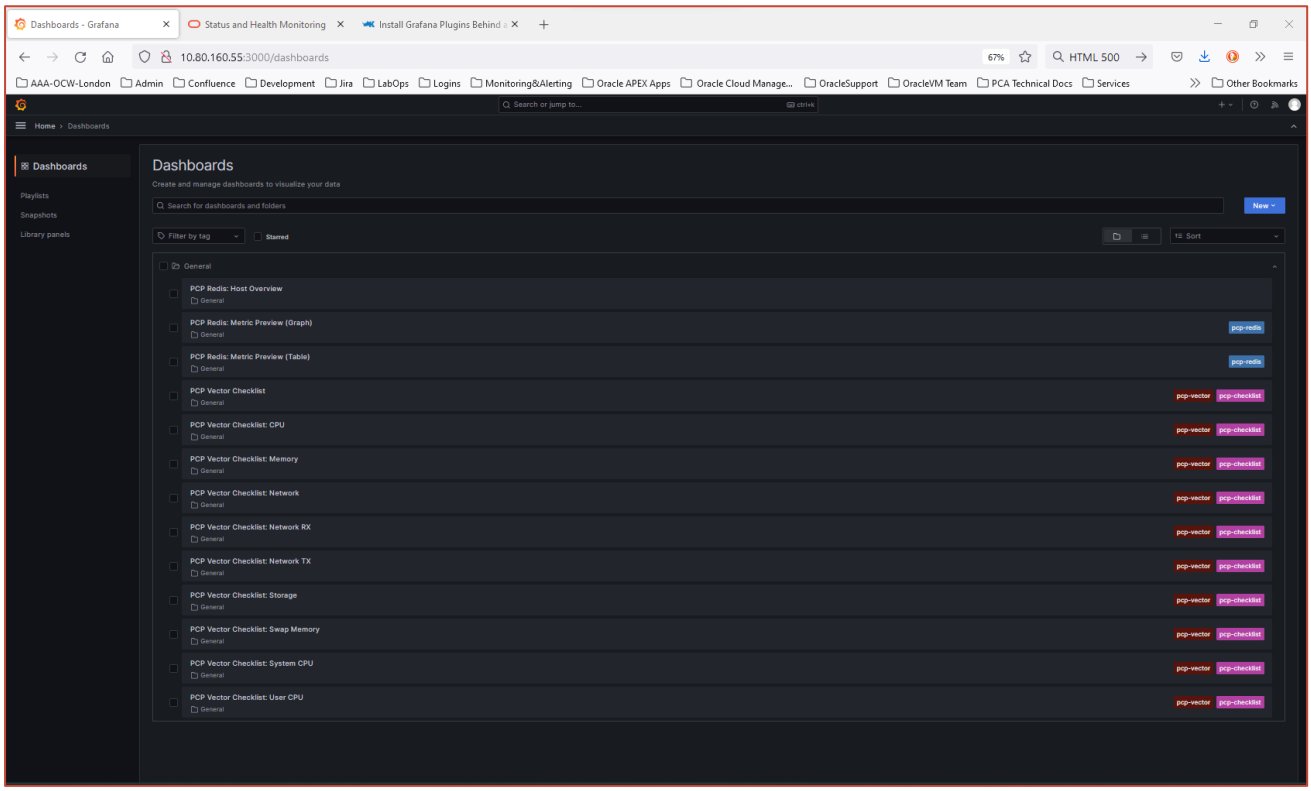
With a working PCP Redis Data Source, access to the collated metrics is now possible.

From the Grafana Home Page Menu Bar, access the available Dashboards:



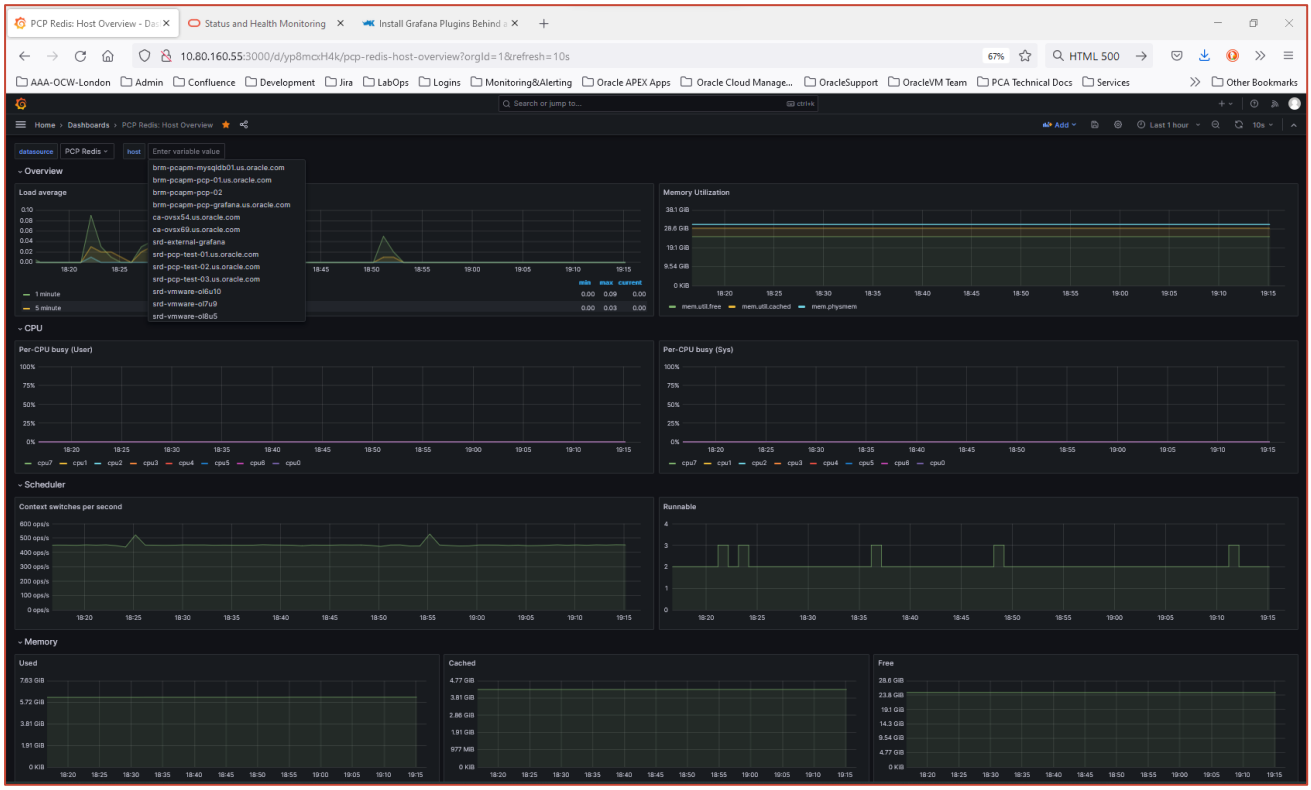
Grafana – Access PCP Redis Dashboards

A list of the currently available Grafana Dashboards (within the General Folder) is displayed:



Grafana – List available PCP Redis Dashboards

In this case, the ‘PCP Redis: Host Overview’ Dashboard will be selected:



Grafana – PCP Redis Host Overview Dashboard

From within the ‘PCP Redis: Host Overview’ Grafana Dashboard, the Host drop down list now displayed the expected thirteen individual systems being collated by the VM Instance Collector.

Next Steps

Now that Grafana has been shown to be able to display the Host level performance metrics being collected and collated, some additional steps are required:

Additional Data Sources

The defined PCP Redis Data Source is for a single PCP Metrics collation service, using the combination of pmlogger, redis and pmproxy as outlined above.

For more complex environments, with multiple PCP Metrics collectors, additional PCP Redis Data Sources will need to be defined for each collation service.

Creating Dashboards

The default Grafana Dashboards provided by the PCP Redis plugin demonstrate the capabilities of this approach. The creation of further customer specific Grafana Dashboards is out of scope for this document.

The Grafana Documentation Library (see below) provides both instructional videos and detailed documentation on how to create new Grafana Dashboards.

Monitoring and Alerting

The PCP Redis plugin is enabled for Alerting. The metrics being presented to Grafana can have alerts associated with metric value thresholds to provide both ‘Warning’ and ‘Error’ status alerts.

The precise mechanism for this is heavily dependent on the version of Grafana being used. Please refer to the Grafana Documentation for the specific version of Grafana deployed.

In summary, this will require the following steps to be completed:

- Create and configure one, or more, Notification Channels or Contact Points – depending on the version of Grafana in use
- Create metric specific Alert Rules, including any required threshold settings, within the Grafana Alerting Framework and associate the Alerts with the appropriate Notification Chanel or Contact Point
- Trigger test alerts to verify the configuration(s) in place

This completes this section of the step-by-step guide.

Section References

The following URL's provide links to additional documentation:

- Performance Co-Pilot Grafana-PCP - https://grafana-pcp.readthedocs.io/en/latest/index.html#
- Grafana Documentation Library – https://grafana.com/docs/grafana/latest/
- Grafana Setup – https://grafana.com/docs/grafana/latest/setup-grafana/
- Grafana Administration – https://grafana.com/docs/grafana/latest/administration/
- Grafana Security – https://grafana.com/docs/grafana/latest/setup-grafana/configure-security/
- Grafana Dashboards – https://grafana.com/docs/grafana/latest/dashboards/
- Grafana Alerting – https://grafana.com/docs/grafana/latest/alerting/

Reference Materials

The following reference URLs provide a consolidated summary of the various section references provided elsewhere within this document:

Oracle References

- Oracle Linux 9 – Reference Library – https://docs.oracle.com/en/operating-systems/oracle-linux/9/
- Oracle Linux 9 – Working with Performance Co-Pilot - https://docs.oracle.com/en/operating-systems/oracle-linux/9/monitoring/monitoring-WorkingWithPerformanceCoPilot.html#pcp
- Oracle Linux 8 – Reference Library – https://docs.oracle.com/en/operating-systems/oracle-linux/8/
- Oracle Linux 8 – Working with Performance Co-Pilot – https://docs.oracle.com/en/operating-systems/oracle-linux/8/monitoring/monitoring-WorkingWithPerformanceCoPilot.html#pcp

Performance Co-Pilot References

- Performance Co-Pilot documentation – https://pcp.readthedocs.io/en/latest/
- Performance Co-Pilot documentation – Quick Reference Guide – https://pcp.readthedocs.io/en/latest/QG/QuickReferenceGuide.html
- Performance Co-Pilot documentation – PCP Archive Logger Deployment – https://pcp.readthedocs.io/en/latest/UAG/PcpDeploymentStrategies.html#pcp-archive-logger-deployment
- Performance Co-Pilot documentation - PCP Redis – https://grafana-pcp.readthedocs.io/en/latest/datasources/redis.html
- Performance Co-Pilot documentation – Troubleshooting – https://grafana-pcp.readthedocs.io/en/latest/troubleshooting.html
- Performance Co-Pilot Grafana-PCP - https://grafana-pcp.readthedocs.io/en/latest/index.html#

Grafana References

- Grafana Document Library – https://grafana.com/docs/grafana/latest/
- Grafana Setup – https://grafana.com/docs/grafana/latest/setup-grafana/
- Grafana Administration – https://grafana.com/docs/grafana/latest/administration/
- Grafana Security – https://grafana.com/docs/grafana/latest/setup-grafana/configure-security/
- Grafana Alerting – https://grafana.com/docs/grafana/latest/alerting/
- Grafana Data Source documentation – https://grafana.com/docs/grafana/latest/datasources/
- Grafana Dashboard Documentation – https://grafana.com/docs/grafana/latest/dashboards/

Other References

- Redis Documentation – https://redis.io/docs/about/

Connect with us

Call +**1.800.ORACLE1** or visit **oracle.com**. Outside North America, find your local office at: **oracle.com/contact**.

blogs.oracle.com facebook.com/oracle twitter.com/oracle