How to obtain and interpret a Cloud Optimizer Operational Status Report

2017/12/19

Thierry Ledent
thierry.ledent@microfocus.com

Get the latest updates of this document at https://softwaresupport.softwaregrp.com/km/KM03049997.

Abstract

The troubleshooting toolkit PVTK 2.0 provides options to generate a Cloud Optimizer Operational Status Report. This report provides extended information on the current status and health of the Cloud Optimizer server and processes. It is often a valuable starting point for troubleshooting Cloud Optimizer problems, or simply for verifying that Cloud Optimizer is operating normally.

This paper explains how to obtain the Cloud Optimizer Operational Status Report and how to interpret it. It highlights the pieces of information that may hint to typical failure scenarios and provides advice for initial troubleshooting steps.

1 Obtaining a CO Operational Status report

To obtain a Cloud Optimizer Operational Status Report, run the below command as root on your CO server:

```
# /opt/OV/contrib/PVTK/pvsupport -status all
```

If you miss this command `pvsupport` on your CO server, then install the PVTK 2.0 troubleshooting toolkit available for download at https://softwaresupport.softwaregrp.com/km/KM02600652.

For detailed usage information, check the PVTK 2.0 User and Reference manual included in the installation package and in the folder `/opt/OV/contrib/PVTK/doc`.

You can also use the PVTK wizard to generate and navigate through an Operational Status Report:

```
# /opt/OV/contrib/PVTK/pvtk -wizard
```

Then select the menu options:

```
3. Show an operational status report
2. Show a complete operational status report
```

Use the keys **Space**, **Up**, **Down**, **PgUp**, **PgDn**, **Home** and **End** to navigate through the report. Use the key **q** to return to the wizard.

If you are already familiar with the `pvsupport -grab` command for collecting a pvsupport archive, note that a CO Operational Status Report is included by default in the archive's folder `SUPPORT`.

2 Interpreting a CO Operational Status Report

The operational status report includes a header and up to 30 sections that describe the status and/or health of one particular aspect of the CO server. Depending on which features and integrations have been configured some sections may be partly empty or skipped when creating the report.

Note also that this discussion describes a complete report, generated with `pvsupport -status all`. Partial reports can also be generated that focus on certain areas only.

2.1 Report header

The report header provides information on when and where the report was generated as well as some version information.

```
pvsupport.sh 2.0
Date:           Mon Dec 11 16:56:18 CET 2017
Hostname:       CO701.gale7.net
OS:             Red Hat Enterprise Linux Workstation release 6.4 (Santiago)
Kernel:         2.6.32-358.el6.x86_64
Uptime:         16:56:18 up 8 days, 23:39,  4 users,  load average: 1.22, 1.47, 1.50
CO version:     HPE Cloud Optimizer Version: 03.02.004 (Installed - not appliance)
CO HF (last):   CO_3.02.004_HF_VCENTER_4
Tomcat version: HPOvTomcatB-7.00.078-1.x86_64
DB version:     vertica-8.1.0-2.x86_64
OA version:     12.02.008
```

It is worth highlighting a few items on this header:

- The first line provides the version information of the command `pvsupport`. Note pvsupport version 1.0x can also generate operational status reports, but with slightly less information.
- The entry "CO version" returns the version displayed by the command `pv version`. It also provides the information between parentheses whether CO was deployed as a virtual appliance or if it was installed on an pre-existing server.
- The entry "CO HF (Last)" returns the hotfix information displayed by the command `pv hfv`. This is the last installed hotfix (not the full list) and may even show an old inactive hotfix if CO was upgraded to a newer version since this last hotfix was installed. For a better understanding of the installed hotfixes, it is best to check the next section.

2.2 CO hotfix history

The next section shows the list of installed hotfixes along with the installation timestamp.

```
=================
CO hotfix history
=================
drwxrwsr-x 2 root cogroup 4096 Feb 16  2016 02.20.388_4
drwxrwsr-x 2 root cogroup 4096 Jan 18  2017 02.20.388_5
drwxr-sr-x 2 root cogroup 4096 Oct  4 10:19 CO_3.02.004_HF_VCENTER_3
drwxr-sr-x 2 root cogroup 4096 Oct  4 14:55 CO_3.02.004_HF_VCENTER_4
```

As shown above, the list also includes hotfixes that were installed before CO was upgraded to its current version.

2.3 OA component versions

This section is available only if an OA instance (Operations Agent) has been installed on the CO server. It provides the output of the command `ovdeploy -inv`.

```
======================
OA components versions
======================
NAME              DESCRIPTION                                VERSION    TYPE OSTYPE
HPOvAgtLc         HPE Operations agent L10N Package          12.04.006 pkg  Linux
HPOvBbc           HPE Software HTTP Communication            12.04.006 pkg  Linux
```

```
HPOvConf          HPE Software Configuration                  12.02.008 pkg  Linux
HPOvCtrl          HPE Software Process Control                12.04.006 pkg  Linux
HPOvDepl          HPE Software Deployment                     12.02.008 pkg  Linux
HPOvEaAgt         HPE Software E/A Agent                      12.02.008 pkg  Linux
HPOvGlanc         HPE Software Performance Glance             12.02.008 pkg  Linux
HPOvJPacc         HPE Software Java Performance Access        12.04.006 pkg  Linux
HPOvJbbc          HPE Software HTTP Communication Java        12.04.006 pkg  Linux
HPOvJsec          HPE Software Security Core Java             12.04.006 pkg  Linux
HPOvJxpl          HPE Software Cross Platform Component Java  12.04.006 pkg  Linux
HPOvPacc          HPE Software Performance Access             12.02.008 pkg  Linux
HPOvPerfAgt       HPE Software Performance Agent              12.02.008 pkg  Linux
HPOvPerfMI        HPE Software Measurement Interface          12.02.008 pkg  Linux
HPOvPerlA         HP Software Perl                            05.16.016 pkg  Linux
HPOvSecCC         HPE Software Certificate Management Client  12.04.006 pkg  Linux
HPOvSecCo         HPE Software Security Core                  12.04.006 pkg  Linux
HPOvXpl           HPE Software Cross Platform Component       12.04.006 pkg  Linux
Operations-agent  HPE Operations agent Product                12.02.008 bdl  linux
```

The components highlighted in cyan above are delivered by Cloud Optimizer. The components highlighted in yellow are delivered by Operations Agent. The components highlighted in green are delivered by both Cloud Optimizer and Operations Agent. The version of these components will be the highest of CO or OA.

Note that CO 3.01 and higher currently only supports co-existence with Operations Agent versions 12.00 and higher. Check the support matrix for the most up to date compatibility information: https://softwaresupport.softwaregrp.com/km/KM323488.

2.4 Processes status

This sections lists the running Cloud Optimizer, Vertica and Operations Agent processes along with their PID.

```
================
Processes status
================
ovtomcatB   OV Tomcat(B) Servlet Container      WEB,SERVER    (3848)   Running
pvcd        PV Core                             PV            (3595)   Running
midaemon    Measurement Interface daemon                      (2663)   Running
ttd         ARM registration daemon                           (2780)   Running
perfalarm   Alarm generator                                   Stopped
perfd       real time server                                  (3028)   Running
agtrep      OV Discovery Agent                  AGENT,AgtRep  (4359)   Running
hpcsrvd     HPCS Server                         AGENT,OA      (4309)   Running
hpsensor    HP Compute Sensor                   AGENT,OA      (2647)   Running
oacore      Operations Agent Core               AGENT,OA      (2857)   Running
ompolparm   OM Parameter Handler                AGENT,EA      (4274)   Running
opcacta     OVO Action Agent                    AGENT,EA      (4416)   Running
opcgeni     Generic Source Interceptor          AGENT,EA      (515826) Running
opcmona     OVO Monitor Agent                   AGENT,EA      (4266)   Running
opcmsga     OVO Message Agent                   AGENT,EA      (4307)   Running
opcmsgi     OVO Message Interceptor             AGENT,EA      (4386)   Running
ovbbccb     OV Communication Broker             CORE          (2832)   Running
ovcd        OV Control                          CORE          (2824)   Running
ovconfd     OV Config and Deploy                COREXT        (3992)   Running
Agent Health Status: OK, Time:Sun Dec 10 17:38:20 2017

Message Agent is not buffering.

VIdaemon is running with PID 4228

PV python processes:

Vertica processes:
  3430 /opt/vertica/spread/sbin/spread -c
/var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog/spread.conf -D /opt/vertica/spread/tmp
  3432 /opt/vertica/bin/vertica -D /var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog -C pv -n
v_pv_node0001 -h 127.0.0.1 -p 5433 -P 4803 -Y ipv4 -c
  3454 /opt/vertica/bin/vertica-udx-zygote 15 3 3432 debug-log-off
/var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog/UDxLogs 60 16 0
  4640 /bin/bash /opt/vertica/agent/agent.sh /opt/vertica/config/users/pv_vertica/agent.conf
  4655 /opt/vertica/oss/python/bin/python ./simply_fast.py
```

The following processes are of particular interest:

ovtomcatB	Implements the user interface services. The process ovtomcatB writes its logs and traces to the files /var/opt/OV/log/pvtrace.0.txt and /var/opt/OV/log/tomcat/ovtomcatb.out. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 3. Step -trace appserver.
pvcd	The main Cloud Optimizer process pvcd coordinates all activities, centralizes all collected data, writes it to the database and dispatches it to the user interface and CO alerting scripts. This process writes logs to /var/opt/OV/log/System.txt. It supports XPL tracing and consoleprint tracing. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select options 6. Step -trace pvcd and 5. Step -trace consoleprint.
agtrep	The discovery agent agtrep is an Operations Agent process that executes the policy vPV-Discovery and forwards CI definitions to the OMi server. This process writes logs to /var/opt/OV/log/System.txt and supports XPL tracing. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 9. Step -trace xpl.
hpcsrvd	The compute sensor server hpcsrvd keeps track of the compute sensor agents deployed through the environment. This process writes logs to /var/opt/OV/shared/server/hpcsrv/hpcsrvtrace.log. Logging levels are defined in the file /var/opt/OV/shared/server/hpcsrv/hpcsrv.conf.
opcmona	The monitor agent opcmona is an Operations Agent process that executes the policy vPV-EventMonitor and forwards matched events to the message agent. This process writes logs to /var/opt/OV/log/System.txt and supports XPL tracing. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 9. Step -trace xpl.
opcmsga	The message agent opcmsga is an Operations Agent process that forwards VMware events and CO alerts received from the monitor agent and message interceptor to the OMi server. This process writes logs to /var/opt/OV/log/System.txt and supports XPL tracing. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 9. Step -trace xpl.
opcmsgi	The message interceptor opcmsgi is an Operations Agent process that forwards matched CO alerts to the message agent. This process writes logs to /var/opt/OV/log/System.txt and supports XPL tracing. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 9. Step -trace xpl.
ovbbccb	The communication broker ovbbccb keeps a registry of available services, listens for incoming connections and dispatches the connections to the correct process. This process writes logs to /var/opt/OV/log/System.txt and supports XPL tracing. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 9. Step -trace xpl.
ovcd	The process control daemon ovcd starts, stops and monitors most other processes. The command line interface to ovcd is /opt/OV/bin/ovc. This process writes logs to /var/opt/OV/log/System.txt and supports XPL tracing. For more information on tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 9. Step -trace xpl.
VIdaemon	The VIdaemon is a java process that collects metrics from vCenters. It runs only if vCenter targets are configured. Note the PID of this process (4228), as you will need it when analyzing section 2.9 below. This process writes logs to /var/opt/OV/log/status.virtserver. For tracing options, run the command /opt/OV/contrib/PVTK/pvsupport -help trace and select option 8. Step -trace vidaemon.
vertica	The main database process is vertica. The database logs can be found in: /var/opt/OV/databases/pv/catalog/pv/dbLog /var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog/vertica.log

As shown in above example, it is common that the process perfalarm is stopped. This depends essentially on the type of Operations Agent license that is installed and has no impact on Cloud Optimizer, nor any integration with Cloud Optimizer.

2.5 Target/Collection status and configuration

This section lists the configured targets with their current collection status and some key configuration settings.

```
========================================
Target/Collection status and configuration
========================================
HYPERV :
   HYPV1\cocoll#HYPV1@hypv1.gale2.net (3)  : Data Collection Completed (0:4:20 ago) (1513007522)
VCENTER :
   Administrator@10.0.2.30 (28)  : Data Collection Completed (0:2:42 ago) (1513007620)

Alerts were generated at (23:12:26 ago) (1512924236)

[pvcd.hyperv]Targets=HYPV1\cocoll#HYPV1:none@hypv1.gale2.net:1
[pvcd.vcenter]Targets=Administrator:w3khJfIKFp2HhojX84IOhY5grAsweY62@10.0.2.30:1

[pvcd]CollectionFlag=true
[pvcd]CollectionInterval=300
[pvcd]Virt_Node_Coll_Interval=900   (Not used)
```

The first few lines of this section present the output of the command `pvconfig -lt`. The collection status of each target should alternate between `Data collection In Progess` and `Data Collection Completed` at intervals equal to the collection interval. However, for targets of type `HYPERV`, the status should always show `Data Collection Completed`, because the collection happens remotely so that its progress is not exposed to the CO server.

A collection status such as `Data Collection Failed` or `Slow/Partial Data Collection` indicates that there is a problem with the collection for this target. Many such problems can be resolved by applying the vCenter settings recommended in the knowledge document https://softwaresupport.softwaregrp.com/km/KM02945185.

The status `Not Started` indicates that the collection for this target has never been scheduled since the target was configured. This status should only appear for a very short time after the target was configured.

The time since the collection changed to the current status is given between parentheses (highlighted in green above). This should always be less than the collection interval, else it means that the collection is not scheduled on a regular basis.

The collector logs for vCenter targets are written to `/var/opt/OV/log/status.virtserver`. For the Hyper-V collector, the logs are written to `vPVWinVirtCollector.log` in the installation folder of the collector binaries (check the path in the Windows service `HPE vPV Collector Service`).

If the variable `CollectionFlag` is set to false, then data collection has been globally disabled. Data collection can also be disabled per target. The variables Targets include a `:1` (data collection enabled) or a `:0` (data collection disabled) after each target definition (as highlighted on yellow background in above example output). Collection can be enabled/disabled with the command `pvconfig`.

The collection interval is defined in the variable `CollectionInterval`. Valid values are 300 and 900 (seconds). Any other setting will fall back to the default value 300. The variable `Virt_Node_Coll_Interval` exists for historical reasons only and is ignored.

2.6 Database connection test and node status

This section checks the status of the database and provides key configuration information, such as the location of database logfiles and the IP and port used to connect to the database.

```
========================================
Database connection test and node status
========================================
 List of databases
 name | user_name
------+------------
 pv   | pv_vertica

 Node          | Host      | State | Version         | DB
---------------+-----------+-------+-----------------+----
 v pv node0001 | 127.0.0.1 | UP    | vertica-8.1.0.2 | pv

Database: pv
Database Log: /var/opt/OV/databases/pv/catalog/pv/dbLog,
/var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog/vertica.log
Hosts: 127.0.0.1
Restart Policy: ksafe
Port: 5433
Catalog Directory: /var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog
```

There will be a clear indication if the database and/or vertica node is not responsive, e.g.:

```
========================================
Database connection test and node status
========================================
vsql: could not connect to server: Connection refused
        Is the server running on host "127.0.0.1" and accepting
        TCP/IP connections on port 5433?
 Node          | Host      | State | Version         | DB
---------------+-----------+-------+-----------------+----
 v_pv_node0001 | 127.0.0.1 | DOWN  | vertica-8.1.0.2 | pv

Database: pv
Database Log: /var/opt/OV/databases/pv/catalog/pv/dbLog,
/var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog/vertica.log
Hosts: 127.0.0.1
Restart Policy: ksafe
Port: 5433
Catalog Directory: /var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog
```

There are two common scenarios leading to the database going down:

1. The database may terminate if the file system hosting `/var/opt/OV/databases/pv` runs out of space
2. The kernel will commonly pick the process `vertica` as a candidate process to kill if the server runs out of memory

The latter event can be tracked in `/var/log/messages` (included in the pvsupport archive):

```
Nov 26 19:36:43 co701 kernel: Out of memory: Kill process 223722 (vertica) score 177 or sacrifice child
```

If the database goes down while the processes `pvcd` and/or `ovtomcatB` are running, then these processes must be stopped and restarted after the database is up again. The below command will take care of restarting the database and all necessary processes in the correct order:

```
# /opt/OV/contrib/PVTK/pvsupport -restart
```

2.7 Recent table updates tests

This section checks when was the last update to two important tables of the database.

```
==========================
Recent table updates tests
==========================
dml_inst_Infrastructure__Node:       OK      (last update 175 seconds ago <= 1 collection interval)
dml_inst_Virtualization__Datastore:  OK      (last update 175 seconds ago <= 1 collection interval)
```

The tables `dml_inst_Infrastructure__Node` and `dml_inst_Virtualization__Datastore` should be updated at every collection interval, if any target has been configured. This section will show `WARNING` or `ERROR` if the last update was more than respectively 1 collection interval or 2 collection intervals ago.

If the tables are not updated, although the collection status is normal and the database is responsive, this is usually due to one of below two problems:

- The process `pvcd` lost connection with the database. This can happen if the database was restarted while `pvcd` was running. See also section 2.9 below.
- The file system where `/var/opt/OV/databases/pv` is hosted is running low on free space. In this case, the database may reject transactions to avoid running out of disk space while updating the database files. Note that the database may start rejecting transactions as soon as the free space drops to a few GBs.

The latter event can be tracked in `/var/opt/OV/databases/pv/catalog/pv/v_pv_node0001_catalog/vertica.log`:

```
2017-12-19 12:31:25.411 EEThread:7f4979b61700-a0000003a099f6 [EE] <WARNING> Exception in finalize:
Could not write to [/var/opt/OV/databases/pv/data/pv/v_pv_node0001_data/CpD_51596/000000/000_0.gt]:
Volume [/var/opt/OV/databases/pv/data/pv/v_pv_node0001_data] has insufficient space.

...

017-12-19 12:31:25.457 Init Session:7f4971ffe700-a0000003a099f6 <ERROR> @v_pv_node0001: 53100/2927:
Could not write to [/var/opt/OV/databases/pv/data/pv/v_pv_node0001_data/CpD_51596/000000/000_0.gt]:
Volume [/var/opt/OV/databases/pv/data/pv/v_pv_node0001_data] has insufficient space.
```

2.8 vCenter connectivity tests

This section tests the connectivity to the vCenters and some key vCenter configuration settings.

```
=========================
vCenter connectivity tests
=========================
Testing SSL3 connection to vcw701.gale7.net port 443 : Failed with return code 1
    140423930193736:error:1409E0E5:SSL routines:SSL3 WRITE BYTES:ssl handshake failure:s3 pkt.c:598:
    CONNECTED(00000004)
    ---
    For details, try: /usr/bin/openssl s_client -ssl3 -connect vcw701.gale7.net:443
Testing TLS1 connection to vcw701.gale7.net port 443 : OK

 LastObservedTime        Target        Version    Domain  Status    Configuration     MissingUserPrivileges
------------------- --------------- ------------ ------- ------ -------------------- ---------------------
2017-12-18 15:32:26 vcw701.gale7.net 5.5, 3252642 VCENTER 0      Statistics Level = 2 None

NOTE: The Configuration and MissingUserPrivileges reflect the status as observed
```

In the above example, the connectivity test to the vCenter vcw701 failed when using protocol SSL3 and succeeded when using protocol TLS1. As long as the connectivity test succeeds for one protocol, it is ok.

Note that this connectivity test only establishes a connection; it does not perform a login to the vCenter, so it does not help validating the credentials configured for the target in CO.

The second part extracts information from the database collected by CO about the target, such as the vCenter version (5.5 build 3252642), the collection status (0 = OK), the statistics level (2) and any missing user privileges (none). If the statistics level is lower than 2 and/or some missing user privileges are listed, the collection may be incomplete. Depending on the exact problem, the impact on CO may be limited to a very specific feature.

2.9 pvcd connections

This section displays the established connections involving process `pvcd` and the ports on which `pvcd` is listening.

```
================
pvcd connections
================
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State       PID/Program
name
tcp        0      0 127.0.0.1:5433              127.0.0.1:46332             ESTABLISHED 3432/vertica
tcp        0      0 127.0.0.1:46332             127.0.0.1:5433              ESTABLISHED 3595/pvcd
---
tcp        0      0 127.0.0.1:5433              127.0.0.1:46333             ESTABLISHED 3432/vertica
tcp        0      0 127.0.0.1:46333             127.0.0.1:5433              ESTABLISHED 3595/pvcd
---
tcp        0      0 ::1:33989                   :::*                        LISTEN      3595/pvcd
tcp        0      0 ::1:53651                   ::1:33989                   ESTABLISHED 2832/ovbbccb
tcp        0      0 ::1:53655                   ::1:33989                   ESTABLISHED 2832/ovbbccb
tcp        0      0 ::1:33989                   ::1:53655                   ESTABLISHED 3595/pvcd
tcp        0      0 ::1:33989                   ::1:53654                   ESTABLISHED 3595/pvcd
tcp        0      0 ::1:53654                   ::1:33989                   ESTABLISHED 2832/ovbbccb
tcp        0      0 ::1:53400                   ::1:33989                   ESTABLISHED 2832/ovbbccb
tcp        0      0 ::1:33989                   ::1:53651                   ESTABLISHED 3595/pvcd
tcp        0      0 ::1:33989                   ::1:53400                   ESTABLISHED 3595/pvcd
---
tcp        0      0 ::1:36849                   :::*                        LISTEN      3595/pvcd
---
tcp        0      0 ::1:51960                   ::1:50796                   ESTABLISHED 2832/ovbbccb
tcp        0      0 ::1:50796                   ::1:51960                   ESTABLISHED 3595/pvcd
---
tcp        0      0 ::1:53934                   :::*                        LISTEN      3595/pvcd
tcp        0      0 ::1:45156                   ::1:53934                   ESTABLISHED 2832/ovbbccb
tcp        0      0 ::1:53934                   ::1:45156                   ESTABLISHED 3595/pvcd
---
tcp        0      0 ::1:58068                   ::1:39562                   ESTABLISHED 3595/pvcd
tcp        0      0 ::1:39562                   ::1:58068                   ESTABLISHED 4228/java
---
tcp        0      0 :::50796                    :::*                        LISTEN      3595/pvcd
```

The process `pvcd` should have at least one established connection to the database process `vertica` as highlighted in green in the above example.

If the CO server is configured with one or more vCenter targets, `pvcd` should have one established connection with the java process `VIdaemon` (find the PID in section 2.4 above) as highlighted in yellow above.

If these connections are missing, a restart of all processes can be attempted for a quick resolution:

```
# /opt/OV/contrib/PVTK/pvsupport -restart
```

The process `pvcd` will typically also have several established connections with process `ovbbccb`.

2.10 ovtomcatB connections

This section displays the established connections involving process `ovtcomcatB` and the ports on which `ovtomcatB` is listening.

```
====================
ovtomcatB connections
====================
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State       PID/Program
name
tcp        0      0 :::8010                     :::*                        LISTEN      3848/java
---
tcp        0      0 :::8081                     :::*                        LISTEN      3848/java
---
tcp        0      0 :::8444                     :::*                        LISTEN      3848/java
---
tcp        0      0 127.0.0.1:5053              127.0.0.1:38818             ESTABLISHED 2602/ovtrcd
tcp        0      0 ::ffff:127.0.0.1:38818      ::ffff:127.0.0.1:5053       ESTABLISHED 3848/java
---
tcp        0      0 127.0.0.1:5433              127.0.0.1:46479             ESTABLISHED 3432/vertica
tcp        0      0 ::ffff:127.0.0.1:46479      ::ffff:127.0.0.1:5433       ESTABLISHED 3848/java
---
tcp        0      0 127.0.0.1:5433              127.0.0.1:46481             ESTABLISHED 3432/vertica
tcp        0      0 ::ffff:127.0.0.1:46481      ::ffff:127.0.0.1:5433       ESTABLISHED 3848/java
---
tcp        0      0 ::ffff:127.0.0.1:8006       :::*                        LISTEN      3848/java
```

The process `ovtomcatB` should have at least one established connection to the database process `vertica` as highlighted in green in the above example.

If this connection is missing, a restart of `ovtcomcatB` can be attempted for a quick resolution:

```
# /opt/OV/contrib/PVTK/pvsupport -restart ovtomcatB
```

2.11 Application server input files

This section provides a listing of application server input files.

```
==============================
Application server input files
==============================
-rw-rw-r-- 1 couser cogroup  3229 Dec 11 16:52 /var/opt/OV/conf/perf/HYPERV.xml
-rw-rw-r-- 1 couser cogroup 24108 Dec 11 16:52 /var/opt/OV/conf/perf/VCENTER.xml
```

These files contain data exported from the CO database that is used by process `ovtomcatB` to populate the treemap and parts of the dashboard in the CO user interface.

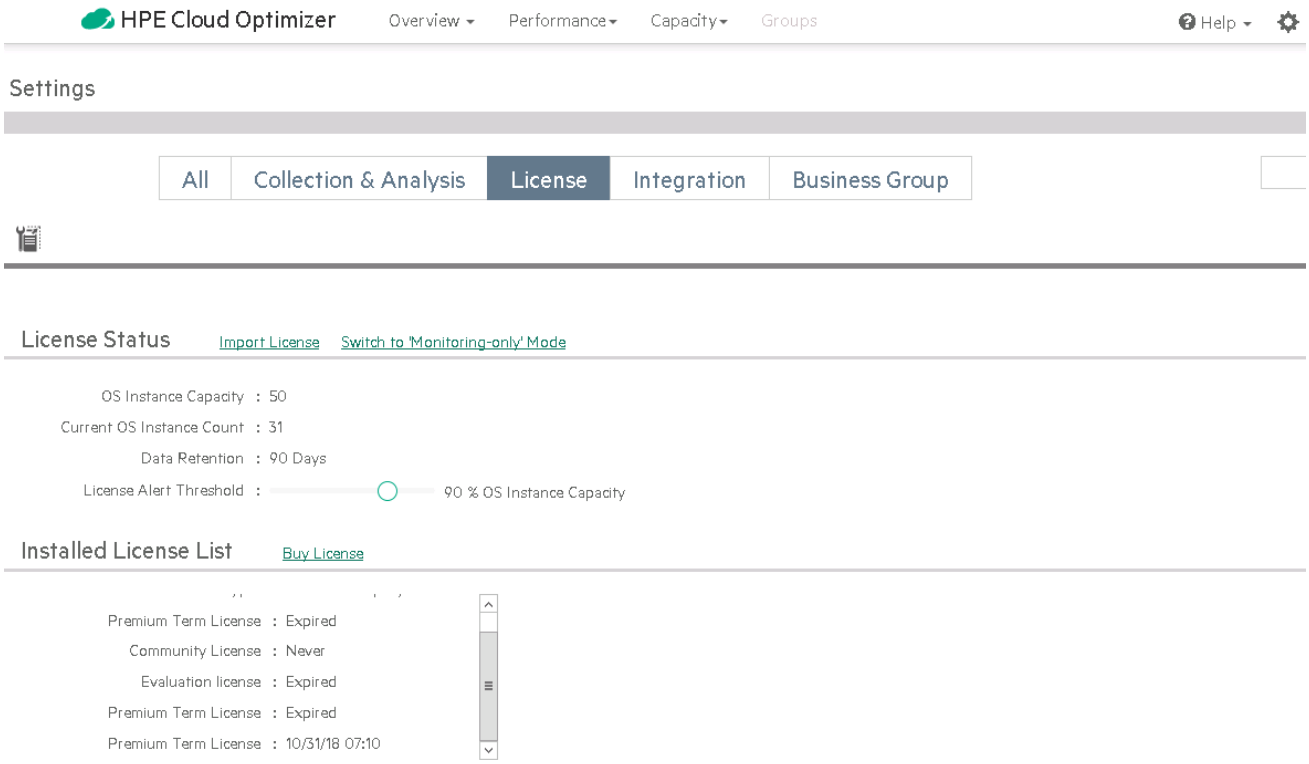
The timestamps of these files should update at the same frequency as the collection interval. If the timestamps are not updating, it usually hints to a problem with the database or the process `ovtomcatB`.

2.12 Licenses

This section provides the output of the command `pv license`.

```
==================
/usr/bin/pv license
==================
License Details ..
License Type      License Expiry Date
Premium Term License    Sun Nov 19 00:59:59 CET 2017
Community License       Never
Evaluation license      Expired
Premium Term License    Thu Jul 06 01:59:59 CEST 2017
Premium Term License    Wed Oct 31 07:10:02 CET 2018
```

Various CO features are enabled only when the proper license is installed. The license should cover the number of collected instances. This can only be verified in the admin page of the CO user interface, e.g.:



2.13 OA policies

This section is available only if an OA instance (Operations Agent) has been installed on the CO server. It provides the output of the command `ovpolicy -list`.

```
===========
OA policies
===========
       * List installed policies for host 'localhost'.

  Type              Name                          Status     Version
 ------------------------------------------------------------------
  configfile        "vPV Alert Sensitivity"       enabled    0302.0000
  configfile        "vPV Custom Alert Sensitivity Definition"   enabled    0302.0000
  configfiletmpl    "vPV Alert Sensitivity"       enabled    0302.0000
  configsettings    "vPV-SuppresAlert"            enabled    0302.0000
  monitor           "vPV-EventMonitor"            enabled    0302.0000
  monitortmpl       "vPV-EventMonitor"            enabled    0302.0000
  msgi              "vPV-OMIntegration"           enabled    0302.0001
  svcdisc           "vPV-Discovery"               enabled    0302.0000
```

The CO/OMi integration uses these policies to forward data from the CO server to the OMi server:

vPV Alert Sensitivity	This policy enables to select the sensitivity level for CO alerts.
vPV Custom Alert Sensitivity Definitions	This policy enables to configure thresholds for the custom sensitivity level for CO alerts.
vPV-SuppressAlert	This policy enables to configure rules for the suppression of CO alerts.
vPV-EventMonitor	This policy forwards collected VMware events to OMi.
vPV-OMIntegration	This policy forwards CO alerts to OMi.
vPV-Discovery	This policy generates and forwards CI definitions to OMi.

2.14 Trace settings

This section lists the logging/tracing settings of several CO components.

```
==============
Trace settings
==============
VIdaemon logfile name:                    /var/opt/OV/log/status.virtserver
VIdaemon logfile settings:                maxFileSize=40960KB, maxBackupIndex=5
VIdaemon com.hp.virt:                     level=info
[pvcd.Framework]CONSOLEPRINT:             false
[pvcd.OAPipeRemoteReceiver]CONSOLEPRINT:  false
[pvcd.OAPipeRemoteSender]CONSOLEPRINT:    false
[pvcd.db]CONSOLEPRINT:                    false
[pvcd.dml]CONSOLEPRINT:                   false
pvcd consoleprint trace file:             /var/opt/OV/tmp/pvcd_consoleprint_<PID_pvcd>.trc
pvcd registration:                        redirection disabled
Analysis trace settings:                  off
Analysis trace file:                      /tmp/analysis.log
Application server trace settings:        TRACELEVEL = 0
Application server trace file:            /var/opt/OV/log/pvtrace.0.txt
53 components XPL trace:                  off
16 alert namespace(s):                    Trace: disabled       TriggeredInstanceList: dropped
```

The above output shows all logging/tracing settings at normal levels.

Higher logging/tracing levels may result into generating large log/trace files. The performance impact of higher logging/tracing levels is usually limited, except:

- The process `pvcd` should only be operating in consoleprint mode when required for troubleshooting purposes. Consoleprint mode is disabled if the settings are as highlighted in green above.
- XPL tracing for process `ovbbccb` should be enabled only when required for troubleshooting purposes. XPL tracing is disabled for a process, if this process is not explicitly listed near the line highlighted in yellow above.

To restore default logging/tracing settings, run the command:

```
# /opt/OV/contrib/PVTK/pvsupport -script reset_all_traces.pvs
```

2.15 Communication with OM server

This section tests the communication with the OMi server.

```
============================
Communication with OM server
============================

[sec.core.auth]MANAGER=OMIL701.gale7.net

# /usr/bin/timeout 20 /opt/OV/bin/bbcutil -ping http://OMIL701.gale7.net

http://OMIL701.gale7.net: status=eServiceOK
                          coreID=d2afe0d0-77c0-7583-0b6c-b6fe2fa73bab bbcV=
                          appN= appV= conn=9 time=17 ms
```

```
# /usr/bin/timeout 20 /opt/OV/bin/bbcutil -ping https://OMIL701.gale7.net

https://OMIL701.gale7.net: status=eServiceOK
                           coreID=d2afe0d0-77c0-7583-0b6c-b6fe2fa73bab
                           bbcV=12.02.008 appN=ovbbccb appV=12.02.008 conn=9
                           time=84 ms

# /usr/bin/timeout 20 /opt/OV/bin/bbcutil -ping https://OMIL701.gale7.net/com.hp.ov.opc.msgr

https://OMIL701.gale7.net/com.hp.ov.opc.msgr:
          status=eServiceOK coreID=ef1ce9b6-d8f6-4140-ad35-f0c80e179901
          bbcV=12.00.056  appN=OMi appV=10.01.303 conn=0 time=142 ms

# /usr/bin/timeout 20 /opt/OV/bin/bbcutil -ping
https://OMIL701.gale7.net/com.hp.ov.ow.SvcDscSvr/SvcDscSvr

https://OMIL701.gale7.net/com.hp.ov.ow.SvcDscSvr/SvcDscSvr:
          status=eServiceOK coreID=ef1ce9b6-d8f6-4140-ad35-f0c80e179901
          bbcV=12.00.056  appN=com.hp.ov.ow.SvcDscSvr appV=unknown version
          conn=0 time=132 ms
```

The first line displays the FQDN of the primary manager. If a primary manager is configured, the next lines test the communication to this manager. Each of these tests should return the status `eServiceOK`.

The first communication test establishes a HTTP connection to the process `ovbbccb` of the OMi server's GW. If this test fails, it usually indicates a network connectivity issue (e.g. connection blocked by a firewall, no routing...) or a misconfigured port (the GW may use a custom port that needs to be configured on the CO server). If the second test succeeds, while the first test fails, it probably indicates that the firewall was configured to only allow HTTPS communication from the CO server to the OMi server. This is OK.

The second communication test established a HTTPS connection to the process `ovbbccb` of the OMi server's GW. If this test fails, while the first test succeeds, it typically indicates that the CO server has not received the proper certificates from the OMi server.

The third communication test establishes a HTTPS connection to the message receiver service of the OMi server. If this test fails, while the second test succeeds, it hints to a problem with the process `WDE` on the GW. Forwarding of CO alerts and VMware events will be impacted.

The fourth communication test establishes a HTTPS connection to the discovery server service of the OMi server. If this test fails, while the second test succeeds, it hints to a problem with the discovery server service. Forwarding of CI information will be impacted.

2.16 ovbbccb

This section provides configuration information for the process `ovbbccb` and a list of established connections.

```
=======
ovbbccb
=======
# /opt/OV/bin/ovbbccb -verbose -listovrg

NOTE:    Sending RPC request to: 'https://localhost:1383/com.hp.ov.bbc.cb/
         bbcrpcserver'.

         NOTE:   HP OpenView resource groups on node 'localhost':

         <default>


# /opt/OV/bin/ovbbccb -verbose -status

NOTE:    Sending status request to: 'https://localhost:1383/Hewlett-Packard/
         OpenView/BBC/status/'.


Status: OK


(Namespace, Port, Bind Address, Open Sockets)

  <default>    1383     ANY    7
```

```
HP OpenView HTTP Communication Incoming Connections

  BBC 12.04.006 ; BBC Application unknown version
          ::1.39818                ::1.1383
  BBC 12.04.006; bbc.http.ext.ctrl 00.00.000
          ::1.37658                ::1.1383
  BBC 12.04.006;  18
          ::1.37653                ::1.1383
  BBC 12.04.006; ovbbccb 12.04.006
          ::1.58519                ::1.1383
  BBC 12.04.006; bbc.http.ext.ctrl 00.00.000
          ::1.41797                ::1.1383
  BBC 12.00.056 ; BBC Application unknown version
          10.7.3.1.53030           10.7.3.10.1383
  BBC 12.04.006;  18
          ::1.37588                ::1.1383
```

The process `ovbbccb` is responsible for accepting most incoming connections and redirecting them to the proper local process. However, incoming connections for the CO user interface are established directly to the `ovtomcatB` process and do not involve `ovbbccb`.

The process `ovbbccb` listens on port 383 by default, but can be configured to listen on a different port, such as 1383 in the example above. The firewall should accept incoming connections to this port.

The last part of this section shows a list of incoming connections. This may show a very dynamic list of connections, typically originating from local processes, from the OMi server, the Hyper-V collectors and/or the OBR server (if integrated with OBR).

2.17 ifconfig

This section provides the output of `ifconfig -a`.

```
==============
/sbin/ifconfig
==============
eth1      Link encap:Ethernet  HWaddr 00:0C:29:F3:68:5B
          inet addr:10.7.3.10  Bcast:10.7.255.255  Mask:255.255.0.0
          inet6 addr: fe80::20c:29ff:fef3:685b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:114815 errors:0 dropped:0 overruns:0 frame:0
          TX packets:96143 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12347199 (11.7 MiB)  TX bytes:10016364 (9.5 MiB)

eth2      Link encap:Ethernet  HWaddr 00:0C:29:F3:68:65
          inet addr:10.0.2.128  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fef3:6865/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2588504 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2509356 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1288742699 (1.2 GiB)  TX bytes:538745092 (513.7 MiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:17494082 errors:0 dropped:0 overruns:0 frame:0
          TX packets:17494082 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:5173387631 (4.8 GiB)  TX bytes:5173387631 (4.8 GiB)

virbr0    Link encap:Ethernet  HWaddr 52:54:00:92:96:FD
          inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

2.18 Local port connectivity tests

This section tests local port connectivity.

```
============================
Local port connectivity tests
============================

NOTE: This tests local connectivity to CO701.gale7.net:<port> and bypasses the firewall(s).
      A successful connection does not imply that a remote server can connect.
      A failed connection implies that a remote server is likelly to fail connecting too.

Port 8444 (HTTPS to console):   Connection to CO701.gale7.net 8444 port [tcp/pcsync-http] succeeded!
Port 8081 (HTTP to console):    Connection to CO701.gale7.net 8081 port [tcp/tproxy] succeeded!
Port 1383 (BBC):                Connection to CO701.gale7.net 1383 port [tcp/gwha] succeeded!
Port 5433 (Vertica):            Connection to CO701.gale7.net 5433 port [tcp/pyrrho] succeeded!
Port 9092 (Health status):      connection failed
Port 5480 (VAMI):               connection failed
```

Since this test runs locally, it doesn't enable to verify that ports are open on the firewall. However it still enables to confirm if a process is listening on the given ports.

The minimum requirement is for the "HTTPS to console" and the "BBC" ports to be responsive (8444 and 1383 in above example).

The Vertica port 5433 is tested here for historical reasons (and the port number is hardcoded which may not be correct). In PVTK 2.0, it is best to validate database connectivity and responsiveness in sections 2.6, 2.7, 2.9 and 2.10 described higher in this paper.

The VAMI port (5480) is only relevant for the CO appliance. The VAMI is used essentially for upgrading CO and the VAMI process are stopped by default. They can be started with the command:

```
# /opt/OV/contrib/PVTK/pvsupport -start vami
```

2.19 iptables

This section provides the output of `iptables -L -n`.

```
====================
/sbin/iptables -L -n
====================
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:53
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:53
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp dpt:67
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:67
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:443
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:135
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:135
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:1383
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:1383
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:35357
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:35357
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:383
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:383
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:5433
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:5433
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:5671
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:5671
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:8081
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:8081
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:8444
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:8444
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:8774
ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           state NEW udp dpt:8774
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited
```

```
Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  0.0.0.0/0            192.168.122.0/24    state RELATED,ESTABLISHED
ACCEPT     all  --  192.168.122.0/24     0.0.0.0/0
ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-port-unreachable
REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

The firewall should allow incoming connections to following ports:

22	This port is used by the OBR integration.
383	This port is used by the Hyper-V collector, the OMi server, the OBR integration and potentially other integrated software to establish connections the communication broker `ovbbccb`. This port is configurable (see section 2.16).
5480 5488 5489	These ports are used by the VAMI interface and relevant only for the CO appliance. The VAMI interface is used essentially for upgrading CO.
8081	This port is used to open the CO user interface over an HTTP connection to process `ovtomcatB`. Note that this connection is automatically redirected to HTTPS.
8444	This port is used to open the CO user interface over an HTTPS connection to process `ovtomcatB`.

2.20 Top CPU consuming processes

This section shows the top 15 CPU consuming processes.

```
============================
Top CPU consuming processes
============================
top - 13:31:07 up 16 days, 23:12,  1 user,  load average: 4.76, 3.10, 2.51
Tasks: 131 total,   1 running, 130 sleeping,   0 stopped,   0 zombie
Cpu(s): 55.9%us, 17.4%sy,  3.3%ni, 22.0%id,  0.4%wa,  0.1%hi,  1.0%si,  0.0%st
Mem:   3924412k total,  3720704k used,   203708k free,   130764k buffers
Swap:  8388604k total,   494636k used,  7893968k free,   894656k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 6583 root      20   0 3130m  18m  10m S 61.7  0.5   0:00.34 java
   12 root      20   0     0    0    0 S  1.8  0.0 216:58.03 events/1
 4800 pv_verti  20   0 3402m 960m  21m S  1.8 25.1   4782:26 vertica
 6574 root      20   0 15020 1168  860 R  1.8  0.0   0:00.13 top
    1 root      20   0 19356 1064  872 S  0.0  0.0   0:07.38 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.02 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   5:55.04 migration/0
    4 root      20   0     0    0    0 S  0.0  0.0   5:45.10 ksoftirqd/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 stopper/0
    6 root      RT   0     0    0    0 S  0.0  0.0   4:55.23 watchdog/0
    7 root      RT   0     0    0    0 S  0.0  0.0  11:41.83 migration/1
    8 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 stopper/1
    9 root      20   0     0    0    0 S  0.0  0.0   7:01.22 ksoftirqd/1
   10 root      RT   0     0    0    0 S  0.0  0.0   4:59.64 watchdog/1
   11 root      20   0     0    0    0 S  0.0  0.0  31:09.08 events/0
```

The top CPU consuming processes are generally the process `java` running the `VIdaemon` (if a vCenter target is configured) and the process `vertica`. These processes may show up in the top positions the majority of the time, often consuming double digit CPU percentages. Over a period of one collection interval, they should however occasionally drop to a single digit number. A few other processes may show up near the top occasionally, such as `pvcd` and `python`.

If the Operations Agent is installed, some of its processes may also occasionally appear near the top.

2.21 Top CPU consuming threads

This section shows the top 15 CPU consuming threads.

```
=========================
Top CPU consuming threads
=========================
top - 13:31:08 up 16 days, 23:12,  1 user,  load average: 4.76, 3.10, 2.51
Tasks: 485 total,   4 running, 481 sleeping,   0 stopped,   0 zombie
Cpu(s): 55.9%us, 17.4%sy,  3.3%ni, 22.0%id,  0.4%wa,  0.1%hi,  1.0%si,  0.0%st
Mem:   3924412k total,  3726300k used,   198112k free,   130764k buffers
Swap:  8388604k total,   494636k used,  7893968k free,   894656k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 6584 root      20   0 3195m  24m  10m R 52.1  0.6   0:00.62 java
 6592 root      20   0 3195m  24m  10m R 31.9  0.6   0:00.23 java
  711 root      20   0     0    0    0 R 23.5  0.0   1102:13 kjournald
 6591 root      20   0 3195m  24m  10m S 13.4  0.6   0:00.09 java
 6597 root      20   0 15288 1476  860 R  6.7  0.0   0:00.11 top
 5209 pv_verti  20   0 3402m 960m  21m S  3.4 25.1 420:42.90 vertica
  989 root      20   0 73752 2340 2012 S  1.7  0.1 156:04.71 vmtoolsd
 5208 pv_verti  20   0 3402m 960m  21m D  1.7 25.1 661:40.43 vertica
 5339 pv_verti  20   0 3402m 960m  21m S  1.7 25.1  58:08.87 vertica
 5341 pv_verti  20   0 3402m 960m  21m S  1.7 25.1 128:30.48 vertica
 7611 root      20   0 4105m 779m  11m S  1.7 20.3 177:26.54 java
    1 root      20   0 19356 1064  872 S  0.0  0.0   0:07.38 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.02 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   5:55.04 migration/0
    4 root      20   0     0    0    0 S  0.0  0.0   5:45.10 ksoftirqd/0
```

This data can be used to drill down into the CPU usage of individual threads when an excessive CPU consumption of a particular process is suspected. It can help distinguish between a looping thread and a thread leak. Some other data is however required for a detailed analysis such as the output of `ps -eLf` and a stacktrace (both available in the pvsupport archive).

2.22 Top memory consuming processes

This sections shows the top 15 memory consuming processes.

```
==============================
Top memory consuming processes
==============================
top - 13:31:08 up 16 days, 23:12,  1 user,  load average: 4.76, 3.10, 2.51
Tasks: 131 total,   1 running, 130 sleeping,   0 stopped,   0 zombie
Cpu(s): 55.9%us, 17.4%sy,  3.3%ni, 22.0%id,  0.4%wa,  0.1%hi,  1.0%si,  0.0%st
Mem:   3924412k total,  3722688k used,   201724k free,   130776k buffers
Swap:  8388604k total,   494636k used,  7893968k free,   894780k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 4800 pv_verti  20   0 3402m 960m  21m S  1.9 25.1   4782:26 vertica
 7501 root      20   0 4105m 779m  11m S  1.9 20.3 708:35.25 java
 3640 root      20   0 3587m 598m  11m S  0.0 15.6 459:09.38 java
 6114 root      20   0 1218m  49m 9548 S  0.0  1.3 193:13.43 pvcd
 3688 root      20   0  629m  47m 7400 S  0.0  1.2 512:51.45 oacore
 3664 root      20   0  820m  21m 9548 S  0.0  0.6 148:44.10 agtrep
 6608 root      20   0 3130m  21m  10m S 114.3  0.5   0:00.59 java
 3670 root      20   0  506m  14m 7108 S  0.0  0.4 127:34.57 opcmona
 6654 pv_verti  20   0 1237m  13m 1744 S  1.9  0.4   1091:16 python
 3712 root      20   0 1145m  10m 6140 S  0.0  0.3 564:44.44 hpsensor
 1230 root      20   0 1154m  10m 5488 S  1.9  0.3  98:27.81 ovcd
 1238 root      20   0  664m 9584 6188 S  0.0  0.2  33:14.24 ovbbccb
 3114 root      20   0 1075m 9508 6916 S  0.0  0.2  41:22.25 opcmsga
 3105 root      20   0  223m 7048 5920 S  0.0  0.2   0:47.90 opcmsgi
 3090 root      20   0  719m 6976 5896 S  0.0  0.2   0:03.16 ovconfd
```

Again the processes `vertica`, `java` (`VIdaemon` and `ovtomcatB`) and `pvcd` will typically show up at the top, usually consuming several GB virtual memory (column `VIRT`). The numbers depend considerably on the size of the monitored environment and may exceed 10GB in large environments.

If the Operations Agent is installed, some of its processes may also appear near the top.

2.23 Memory utilization

This section provides the output of `/usr/bin/free -t`.

```
================
/usr/bin/free -t
================
             total       used       free     shared    buffers     cached
Mem:       4049096    3531392     517704          0      85724    1187344
-/+ buffers/cache:    2258324    1790772
Swap:      4095992     625584    3470408
Total:     8145088    4156976    3988112
```

The most important number here is the amount of free total memory highlighted in green above. If this number drops to zero, the CO server will fail in random ways. One common consequence is that the kernel will select a process to kill, so as to free up some memory for other processes. The database process `vertica` is often selected. See section 2.6.

If the amount of physical memory is too low, the server may experience a bad performance. This can only be identified through a deeper analysis by looking at the amount of paging activity. For instance, with the command:

```
# /usr/bin/sar -B 10 30
```

If this shows a continuous rate of `majflt/s`, and no processes are consuming an abnormal amount of memory, then the system can probably benefit from additional physical memory. A detailed analysis may also point to other tuning options.

2.24 Swap utilization

This section provides the output of `/sbin/swapon -s`.

```
================
/sbin/swapon -s
================
Filename                                Type            Size    Used    Priority
/dev/dm-1                               partition       4095992 625584  -1
```

This information is complementary to section 2.23 and shows how the swap space is distributed across devices.

2.25 File system space utilization

This section provides the output of `/bin/df -Tk`.

```
============
/bin/df -Tk
============
Filesystem    Type   1K-blocks      Used Available Use% Mounted on
/dev/mapper/vg_rhel64x64-lv_root
              ext4    47066056   9029092  35646132  21% /
tmpfs        tmpfs     2024548       224   2024324   1% /dev/shm
/dev/sda1     ext4      495844     38145    432099   9% /boot
/dev/sdb      ext4    51606140   5133960  43850740  11% /var/opt
.host:/     vmhgfs   4294967295 1261792923 3033174372  30% /mnt/hgfs
```

The CO, Vertica and OA binaries are located in `/opt/OV` and `/opt/vertica`. The CO, Vertica and OA data is located in `/var/opt/OV`. Note that the database will start rejecting transactions when the free space in the file system that hosts the database files (`/var/opt/OV/databases/pv`) drops to a few GBs (see also section 2.7), well before the file system reaches 100% utilization. PVTK 2.0 provides a file system cleanup wizard that can help identify, archive or delete disused files. Start the file system cleanup wizard with:

```
# /opt/OV/contrib/PVTK/pvtk -wizard fsclean
```

2.26 File system inode utilization

This section provides the output of `/bin/df -Ti`.

```
===========
/bin/df -Ti
===========
Filesystem    Type    Inodes   IUsed   IFree IUse% Mounted on
/dev/mapper/vg_rhel64x64-lv_root
              ext4   2990080  258414 2731666    9% /
tmpfs        tmpfs    506137       5  506132    1% /dev/shm
/dev/sda1     ext4    128016      39  127977    1% /boot
/dev/sdb      ext4   3276800    3382 3273418    1% /var/opt
.host:/     vmhgfs         0       0       0    -  /mnt/hgfs
```

An inode holds file metadata such as timestamps and file data location in the file system. A file system that runs out of inodes can no longer create new files, even if `df -Tk` still reports free space. This situation can occur when a process creates large amounts of temporary files and omits to delete them. PVTK 2.0 provides a file system cleanup wizard that can help you identify, archive or delete disused files. Start the file system cleanup wizard with:

```
# /opt/OV/contrib/PVTK/pvtk -wizard fsclean
```

2.27 Mount points

This section provides the output of `/bin/mount`.

```
==========
/bin/mount
==========
/dev/mapper/vg_rhel64x64-lv_root on / type ext4 (rw)
proc on /proc type proc (rw)
sysfs on /sys type sysfs (rw)
devpts on /dev/pts type devpts (rw,gid=5,mode=620)
tmpfs on /dev/shm type tmpfs (rw)
/dev/sda1 on /boot type ext4 (rw)
/dev/sdb on /var/opt type ext4 (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
.host:/ on /mnt/hgfs type vmhgfs (rw,ttl=1)
vmware-vmblock on /var/run/vmblock-fuse type fuse.vmware-vmblock
(rw,nosuid,nodev,default_permissions,allow_other)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
gvfs-fuse-daemon on /root/.gvfs type fuse.gvfs-fuse-daemon (rw,nosuid,nodev)
```

2.28 Inter-process communication resources limits

This section provides the output of `/usr/bin/ipcs -l`.

```
================
/usr/bin/ipcs -l
================

------ Shared Memory Limits --------
max number of segments = 4096
max seg size (kbytes) = 67108864
max total shared memory (kbytes) = 17179869184
min seg size (bytes) = 1

------ Semaphore Limits --------
max number of arrays = 128
max semaphores per array = 250
max semaphores system wide = 32000
max ops per semop call = 32
semaphore max value = 32767

------ Messages: Limits --------
max queues system wide = 7908
max size of message (bytes) = 65536
default max size of queue (bytes) = 65536
```

Cloud Optimizer and Operations Agent do not document any requirements for IPC settings (shared memory, semaphores and messages). Shortage of IPC resources is rather uncommon, but could happen, especially if many OA features are used intensively. This section provides the configured limits for IPC settings. The next section provides an overview of the consumed IPC resources.

2.29 Inter-process communication resources utilization

This section provides the output of `/usr/bin/ipcs -u`.

```
================
/usr/bin/ipcs -u
================

------ Shared Memory Status --------
segments allocated 34
pages allocated 1719
pages resident  429
pages swapped   440
Swap performance: 0 attempts     0 successes

------ Semaphore Status --------
used arrays = 25
allocated semaphores = 32

------ Messages: Status --------
allocated queues = 0
used headers = 0
used space = 0 bytes
```

Cloud Optimizer and Operations Agent do not document any requirements for IPC settings (shared memory, semaphores and messages). Shortage of IPC resources is rather uncommon, but could happen, especially if many OA features are used intensively. This section provides an overview of the consumed IPC resources, to be compared with the configured limits shown in the previous section.

2.30 SELinux mode

This section provides the output of command `getenforce`.

```
====================
/usr/sbin/getenforce
====================
Disabled
```