Oracle Linux 8

Monitoring and Tuning the System

F24025-28
October 2025

Oracle Linux 8 Monitoring and Tuning the System,

F24025-28

Contents

Preface

Oracle Linux 8: Monitoring and Tuning the System describes the various utilities, features, and services that you can use to monitor system performance, detect performance issues, and improve the performance of various system components.

1

Monitoring the System and Optimizing Performance

Many performance performance diagnostic utilities are available in Oracle Linux 8 and they can be used to monitor and analyze the resource usage of different hardware components. Tracing tools can be used to diagnose performance issues in several devices, processes, or related threads.

Performance issues are often the result of configuration errors. Using a validated configuration that has been pretested for the enabled software, hardware, storage, drivers, and networking components can minimize those errors. A validated configuration incorporates best practices for an Oracle Linux 8 deployment and has undergone real-world testing of the complete stack. Oracle publishes many validated configurations, which are available for download. See the release notes for the Oracle Linux 8 release that you're running for extra recommendations on kernel parameter settings.

To monitor system performance, collect information about system resources and their usage. For better assessment, establish a baseline of acceptable measurements under typical operating conditions. That baseline can then be used as a reference point that can be used to identify frequent memory shortages, spikes in resource usage, and other problems when they occur. Performance monitoring logs can also be used to plan for future growth and model how configuration changes might affect future performance.

For more information about monitoring the use of resources in the system, also see Working With OSWatcher Black Box and Oracle Linux 8: Collecting and Analyzing Metrics With Performance Co-Pilot.

2

Performance and Monitoring Command Reference

This table provides information about various performance and monitoring commands.

Action	Command	Description
Monitor memory usage and fix fragmented memory.	`adaptivemmd`	Adaptive Memory Management Daemon is a user space service that monitors and automatically fixes memory fragmentation on an Oracle Linux system running the Unbreakable Enterprise Kernel (UEK) Release 6 or newer.
Debug read and write operations on block volumes.	`blktrace`	If the `blktrace` package is installed, it can be used to diagnose and troubleshoot problems with block devices.
Review x86_64 CPU power statistics.	`cpupower`	If the `kernel-tools` package is installed, displays CPU power statistics and sets CPU power states.
Review file system disk space usage.	`df -h`	Displays a list of system mounted volumes including their capacity and how much of that capacity is free or in use.
Review messages in the kernel ring buffer.	`dmesg`	Displays device error messages and records of USB device connections after system boot.
Review used and available memory.	`free`	Displays the total physical and swap memory available and where that memory is allocated.
Open a graphical user interface for reviewing performance data and ending processes.	`gnome-system-monitor`	Starts the graphical system monitor for displaying running processes, memory usage, and mounted file systems.
Monitor block volume I/O activity.	`iostat`	If the `sysstat` package is installed, displays the amount of time that mounted block volumes are active and their average data transfer rates.
Monitor processes creating block volume I/O activity.	`sudo iotop`	If the `iotop` package is installed, displays a list of processes that are performing read and write operations on mounted block volumes.

Action	Command	Description
Review statistics for each networking device.	`ip -s link`	Displays network statistics and errors for all network devices, including packets transmitted (TX) and received (RX). The `dropped` and `overrun` fields provide an indicator of network interface saturation.
Review CPU statistics.	`sudo mpstat`	If the `sysstat` package is installed, displays detailed usage statistics for each CPU.
Monitor block volume I/O activity for NFS mounts.	`nfsiostat`	If the `nfs-utils` package is installed, displays activity statistics for NFS mounts.
Review system activity information.	`sar`	If the `sysstat` package is installed, different options for the `sar` command can be used to display CPU, memory, and file system usage statistics.
Review statistics for each networking protocol.	`ss -s`	Displays network statistics for protocols such as TCP and UDP.
Review running system and user space processes.	`top`	Displays navigable list of processes and their system resource usage.
Review real time reports on x86_64 CPU resource usage.	`turbostat`	If the `kernel-tools` package is installed, displays detailed CPU statistics such as processor frequency, temperature, power usage, and so on.
Review how long the system has been running.	`uptime`	Displays a single line summary of how long the system has been running and load average statistics.
Review memory usage statistics.	`vmstat`	Displays detailed virtual memory statistics.

Monitoring commands that generate a single output can be configured to run on a timed interval for monitoring purposes by using the `watch` command. For example, to run the `mpstat` command every second until it's stopped, run the following command:

```
sudo watch -n 1 mpstat
```

This generates a single-line output that changes information every second, for example:

```
hh:mm:ss  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal
%guest  %gnice   %idle
hh:mm:ss  all    1.44    0.02    0.80    0.01    0.07    0.05    0.06
0.00    0.00   97.56
```

To exit the `watch` command, press the `Ctrl` + `C` keys in combination.

Many of the monitoring commands listed also provide this functionality without needing to use the `watch` command. For example, to run the `mpstat` command every second until it's stopped, run the following command:

```
sudo mpstat 1
```

To verify whether a command provides this option, check its respective manual page.

> **Note**
>
> If Oracle Linux is running in a slim container, or on an Oracle Cloud Infrastructure instance, install the `procps-ng` package to gain access to monitoring tools such as `free`, `top`, and `vmstat`.

Using the Adaptive Memory Management Daemon

Install and run the `adaptivemmd` service.

To manage memory usage, you can use the Adaptive Memory Management daemon, which is available beginning with UEK R6. This daemon is a user space service that monitors free memory on Oracle Linux 8 systems and predicts memory fragmentation and usage. It can also automatically reclaim memory if the system memory becomes too fragmented or is at risk of being filled to capacity.

If the system memory becomes highly fragmented, `adaptivemmd` triggers the kernel to compact memory so that fragmented space can be reclaimed before it's reallocated. If the system is likely to exhaust the available memory then watermarks are adjusted, and this can trigger the kernel to free up new pages in memory. Adaptive Memory Management is available on Oracle Linux 8 systems running the Unbreakable Enterprise Kernel (UEK) Release 6 or newer.

Install the `adaptivemm` package by running the following command:

```
sudo dnf install adaptivemm
```

To enable the `adaptivemmd` service, run the following command:

```
sudo systemctl enable --now adaptivemmd
```

To see the different options that you can use with the adaptivemmd command, use the `-h` option with the `adaptivemmd` command:

```
sudo adaptivemmd -h
```

Configuration options for Adaptive Memory Management Daemon are stored in the `/etc/sysconfig/adaptivemmd` file. For more information, see the `adaptivemmd(8)` manual page.

Using `blktrace` to Debug Block Volume Errors

Use the `blktrace` command to trace and record block volume request operations for troubleshooting purposes.

Install the `blktrace` package by running the following the command:

```
sudo dnf install blktrace
```

Before running the `blktrace` command, mount the kernel level debugging file system:

```
sudo mount -t debugfs debugfs /sys/kernel/debug
```

To output live trace data, use the `blktrace` command with the `blkparse` command. For example, to view the trace data for the `/dev/sda` block volume, run the following command:

```
sudo blktrace -d /dev/sda -o -|blkparse -i -
```

To exit the `blktrace` command, press the `Ctrl` + `C` keys in combination.

For more information about using the `blktrace` command, see the `blktrace(8)` and `blkparse(1)` manual pages.

Using `cpupower` to Control CPU Power States

Review x86_64 CPU statistics and change x86_64 CPU power states by using the `cpupower` command.

Before running the `cpupower` command, install the `kernel-tools` package. This prerequisite is the same regardless of whether Oracle Linux 8 runs on the Unbreakable Enterprise Kernel (UEK) or the Red Hat Compatible Kernel (RHCK).

For more information about the `cpupower` command, see the `cpupower(1)` manual page.

The `cpupower monitor` command can be used to display the current activity level for each CPU that's accessible to the system. To review the default generated report, run the following command:

```
cpupower monitor
```

For more information about options to customize the report output, see the `cpupower-monitor(1)` manual page.

In addition to activity statistics, the previous command also displays statistics for any C-states that have been activated to reduce power consumption for idle CPUs. To review a list of available C-states, run the following command:

```
cpupower idle-info
```

C-states can be enabled and disabled by using the `idle-set` command, and the `-c` option can be used to restrict those changes to a subset of the available CPUs. For more information about C-states and how to enable or disable them, see the `cpupower-idle-info(1)` manual page.

Oracle Linux systems can be optimized for maximum CPU performance, or conversely reduced CPU power consumption, by switching to a different CPU frequency governor. To review the active CPU frequency governor, run the following command:

```
sudo cpupower frequency-info
```

To review a list of available CPU frequency governors, add the `--governors` option to the previous command:

```
sudo cpupower frequency-info --governors
```

To switch to a different CPU frequency governor, use the `cpupower frequency-set` command. For example, to switch to the `performance` CPU frequency governor, run the following command:

```
sudo cpupower frequency-set --governor performance
```

For more information about CPU frequency governors and how to set them, see the `cpupower-frequency-set(1)` manual page.

Using the Graphical System Monitor

Start the graphical system monitor to display information about the system configuration, running processes, resource usage, and file systems.

To display the System Monitor, use the following command:

```
gnome-system-monitor
```

Selecting the `Resources` tab displays the following information:

- CPU usage history in graphical form and the current CPU usage as a percentage.
- Memory and swap usage history in graphical form and the current memory and swap usage.
- Network usage history in graphical form, the current network usage for reception and transmission, and the total amount of data received and transmitted.

To display the System Monitor Manual, press the `F1` key or select `Help`, then select `Contents`.

Using `sysstat` to Monitor and Review System Resource Usage Statistics

Enable the `sysstat` service to collect system logs, then use the `sar`, `mpstat`, `top`, `iostat`, and `iotop` commands to review detailed information about CPU load, memory use, and remaining storage capacity.

Before running the `sar`, `mpstat`, or `iostat` commands, enable the `sysstat` service and its related `systemd` timers so that it generates log entries every 10 minutes while the system is running, and then retains that information for every day of the current month:

```
sudo systemctl enable --now sysstat sysstat-collect.timer sysstat-
summary.timer
```

For more information about running and configuring system services, see Oracle Linux 8: Managing the System With systemd.

The information that the `sysstat` service collects is stored in directory paths following the `/var/log/sa/saDD` pattern. To display the contents of the `sar` log of a specific day of the current month, run the following command:

```
sudo sar -A -f /var/log/sa/saDD
```

You can also use the `sar` command to create a customized log that contains a record of specific information that you want to monitor. Use the following syntax:

```
sudo sar -o datafile seconds count >/dev/null 2>&1 &
```

In the previous command, *datafile* is the full path to the customized log where you want to store the information, while *count* represents the number of samples to record. With this command, the `sar` process runs in the background and collects the data.

To display the results of this monitoring, run the following command, specifying the log file:

```
sudo sar -A -f datafile
```

Reviewing CPU Usage Statistics

To review information about the system CPU load average, run the following command:

```
sar -q
```

```
14:57:55     LINUX RESTART    (2 CPU)

03:00:01 PM   runq-sz  plist-sz   ldavg-1   ldavg-5  ldavg-15   blocked
03:10:01 PM         0       214      0.19      0.30      0.22         0
03:20:01 PM         0       212      0.00      0.05      0.10         0
...
Average:      runq-sz  plist-sz   ldavg-1   ldavg-5  ldavg-15   blocked
Average:            1       212      0.12      0.08      0.05         0
```

The previous command provides more detailed information than the `uptime` command:

```
uptime
```

```
21:25:34 up  6:28,  1 user,  load average: 0.02, 0.10, 0.04
```

To review CPU usage statistics for each CPU core and averaged the data across all the CPU cores, run the following command:

```
sar -u -P ALL
```

```
03:00:01 PM  CPU  %user   %nice   %system   %iowait  %steal   %idle
03:10:01 PM  all  8.30    0.00    2.20      0.22     0.10     89.18
03:10:01 PM    0  8.22    0.00    2.64      0.31     0.09     88.74
03:10:01 PM    1  8.39    0.00    1.77      0.13     0.10     89.61
```

The same information that was collected by the `sysstat` service can also be used with the `mpstat` command to view further detail that was collected about CPU resource usage:

```
mpstat -P ALL
```

```
05:10:01 PM  CPU    %usr   %nice    %sys   %iowait    %irq   %soft  %steal
%guest  %gnice   %idle
05:10:01 PM  all    0.76    0.02    0.70      0.02    0.08    0.05    0.06
0.00    0.00   98.32
05:10:01 PM    0    0.75    0.01    0.68      0.00    0.09    0.03    0.07
0.00    0.00   98.37
05:10:01 PM    1    0.76    0.03    0.72      0.03    0.06    0.06    0.06
0.00    0.00   98.27
```

For a live summary of CPU usage for each running process, and a navigable interface for ending those processes, use the `top` command:

```
top
```

```
top - 22:13:34 up  7:16,  1 user,  load average: 0.00, 0.02, 0.01
Tasks: 149 total,   1 running, 148 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.2 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.2 hi,  0.0 si,  0.0
st
MiB Mem :  14705.5 total,  11738.9 free,    753.2 used,   2213.4 buff/cache
MiB Swap:   4096.0 total,   4096.0 free,      0.0 used.  13588.9 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
34781 root      20   0   11384   7100   1700 S   1.0   0.0   0:00.03 pidstat
 2014 root      20   0   26216   3568   3056 S   0.7   0.0   0:04.78 OSWatcher
 1481 root      20   0  202628  38328  36792 S   0.3   0.3   0:02.94 sssd_nss
...
```

By default, the `top` command output lists the most CPU-intensive processes on the system. The upper section displays general information including the load averages over the past 1, 5, and 15 minutes, the number of running and sleeping processes or tasks, and total CPU, and memory usage.

The second table displays a list of processes, including the process ID number (PID), the process owner, CPU usage, memory usage, running time, and the command name. By default, the list is sorted by CPU usage, with the top consumer of CPU listed first.

To exit the `top` command, press the `Ctrl` + `C` keys in combination.

Reviewing Memory Usage Statistics

To review memory usage statistics, such as free (`kbmemfree`), available (`kbavail`), and used (`kbmemused`) memory, run the following command:

```
sar -r
```

The output also includes `%memused`, which is the percentage of physical memory in use.

To review memory paging statistics, such as page in (`pgpgin/s`) and page out (`pgpgout/s`), page faults and major faults, run the following command:

```
sar -B
```

The output also includes `pgscank/s`, which is the number of memory pages scanned by the `kswapd` daemon each second, and `pgscand/s`, which is the number of memory pages scanned directly each second.

To review swapping statistics, including `pswpin/s` and `pswpout/s`, which are the numbers of pages each second swapped in and out each second, run the following command:

```
sar -W
```

For a full list of options, run the `sar --help` command.

If `%memused` is near 100% and the scan rate is continuously over 200 pages each second, the system has a memory shortage.

When a system runs out of real or physical memory and starts using swap space, system performance deteriorates. If swap space is full then some programs or even the entire OS can malfunction. If the `free` or `top` commands indicate that little swap space remains available, then the system is running low on memory.

Output from the `dmesg` command might also include notifications about problems with physical memory that were detected at boot time.

Reviewing Storage Usage Statistics

To review the number of unused cache entries in the directory cache (`dentunusd`) and the numbers of in-use file handles (`file-nr`), `inode` handlers (`inode-nr`), and pseudo terminals (`pty-nr`), run the following command:

```
sar -v
```

```
12:00:01 AM dentunusd   file-nr  inode-nr    pty-nr
12:10:33 AM     80101      2944     73074         0
12:20:33 AM     79788      2944     72654         0
```

The `iostat` command monitors the loading of block I/O devices by observing the time that the devices are active relative to the average data transfer rates. You can use this information to

adjust the system configuration to balance the I/O loading across disks and host adapters. The following is a sample of the command output:

```
iostat


avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.69    0.05    0.77    0.00    0.03   98.46

Device             tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda               1.05         2.32        25.19     611410    6640659
dm-0              0.62         2.07        20.22     545716    5329660
dm-1              0.70         0.02         4.97       4632    1308788
```

`iostat -x` reports extended statistics about block I/O activity at one second intervals, including `%util`, which is the percentage of CPU time spent handling I/O requests to a device, and `avgqu-sz`, which is the average queue length of I/O requests that were issued to that device. If `%util` approaches 100% or `avgqu-sz` is greater than 1, device saturation is occurring.

Use the `sar -d` command to review similar information on block I/O activity, including values for `%util` and `avgqu-sz`.

If the `iotop` package is installed, the `iotop` utility can be used to identify which processes are responsible for excessive disk I/O. `iotop` has a similar user interface to the `top` command. In its upper section, `iotop` displays the total disk input and output usage in bytes per second. In its earlier section, `iotop` displays I/O information for each process, including disk input output usage in bytes per second, the percentage of time spent swapping in pages from disk or waiting on I/O, and the command name. The following is a sample command output:

```
sudo iotop


Total DISK READ :    0.00 B/s | Total DISK WRITE :       0.00 B/s
Actual DISK READ:    0.00 B/s | Actual DISK WRITE:       0.00 B/s
    TID  PRIO  USER     DISK READ DISK WRITE>    COMMAND
      1 be/4 root        0.00 B/s    0.00 B/s systemd --switched-root --
system --deserialize 16
      2 be/4 root        0.00 B/s    0.00 B/s [kthreadd]
...
```

While reviewing the output, use the `arrow` keys to change the sort field, and press `A` to switch the I/O units between bytes each second and total number of bytes, or `O` to switch between displaying all processes or only those processes that are performing I/O.

To exit the `iotop` command, press the `Ctrl` + `C` keys in combination.

The `nfsiostat` command reports I/O statistics for each NFS file system that's mounted. If this command isn't available, install the `nfs-utils` package.

Using `turbostat` to Review Detailed CPU Statistics

Use `turbostat` to review x86_64 processor frequency, power, and idle state statistics.

Before running the `turbostat` command, install the `kernel-tools` package. This prerequisite is the same regardless of whether Oracle Linux 8 runs on the Unbreakable Enterprise Kernel (UEK) or the Red Hat Compatible Kernel (RHCK).

To monitor CPU statistics and update the output every five seconds, run the following command:

```
turbostat
```

To remove hardware information at the start of the command output, add the `--quiet` option to the `turbostat` command:

```
turbostat --quiet
```

To change the time output interval, use the `-i` option. For example, to see new `turbostat` output every ten seconds, run the following command:

```
turbostat -i 10
```

For more information, see the `turbostat(8)` manual page.

3
Working With the sos Command

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Using sos to Get Debugging Information.

4

Working With TuneD

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Optimize Performance and Power Consumption With TuneD and PowerTOP.

5

Working With OSWatcher Black Box

Oracle OSWatcher Black Box (OSWbb) collects and archives OS and network metrics that you can use to diagnose performance issues. OSWbb operates as a set of background processes on the server and gathers data often, invoking such UNIX utilities as `vmstat`, `mpstat`, `netstat`, `iostat`, and `top`.

OSWbb is useful for Oracle RAC (Real Application Clusters) and Oracle Grid Infrastructure configurations. The RAC-DDT (Diagnostic Data Tool) script file includes OSWbb, but doesn't install it by default.

Installing OSWbb

To install OSWbb:

1. Sign in to My Oracle Support (MOS) at https://support.oracle.com.
2. Download OSWatcher from the link that is listed by Doc ID 301137.1 at https://support.oracle.com/epmos/faces/DocumentDisplay?id=301137.1.
3. Copy the file to the directory that you want to install OSWbb, then run the following command:

   ```
   tar xvf oswbbVERS.tar
   ```

   In the previous command, *VERS* represents the version number of OSWatcher, for example 832 for OSWatcher 8.32.

   Extracting the tar file creates a directory named `oswbb`, which contains all the directories and files that are associated with OSWbb, including the `startOSWbb.sh` script.
4. To enable the collection of `iostat` information for NFS volumes, edit the `OSWatcher.sh` script in the `oswbb` directory, and set the value of `nfs_collect` to `1` as follows:

   ```
   nfs_collect=1
   ```

Running OSWbb

To start OSWbb, run the `startOSWbb.sh` script from the `oswbb` directory.

```
sudo ./startOSWbb.sh [frequency duration]
```

The optional frequency and duration arguments specify how often in seconds OSWbb collects data and the number of hours for which OSWbb runs. The default values are 30 seconds and

48 hours. The following example starts OSWbb recording data at intervals of 60 seconds, and has it record data for 12 hours:

```
sudo ./startOSWbb.sh 60 12


...
Testing for discovery of OS Utilities...
VMSTAT found on your system.
IOSTAT found on your system.
MPSTAT found on your system.
IFCONFIG found on your system.
NETSTAT found on your system.
TOP found on your system.

Testing for discovery of OS CPU COUNT
oswbb is looking for the CPU COUNT on your system
CPU COUNT will be used by oswbba to automatically look for cpu problems

CPU COUNT found on your system.
CPU COUNT = 4

Discovery completed.

Starting OSWatcher Black Box v7.3.0  on date and time
With SnapshotInterval = 60
With ArchiveInterval = 12
...
Data is stored in directory: OSWbba_archive

Starting Data Collection...

oswbb heartbeat: date and time
oswbb heartbeat: date and time + 60 seconds
...
```

In the previous output, *OSWbba_archive* is the path of the archive directory that contains the OSWbb log files.

To stop OSWbb prematurely, run the `stopOSWbb.sh` script from the `oswbb` directory:

```
sudo ./stopOSWbb.sh
```

OSWbb collects data in the directories that are under the `oswbb/archive` directory, which are described in the following table.

Directory	Description
`oswifconfig`	Contains output from `ifconfig`.
`oswiostat`	Contains output from `iostat`.
`oswmeminfo`	Contains a listing of the contents of `/proc/meminfo`.
`oswmpstat`	Contains output from `mpstat`.

Directory	Description
`oswnetstat`	Contains output from `netstat`.
`oswprvtnet`	If you have enable private network tracing for RAC, contains information about the status of the private networks.
`oswps`	Contains output from `ps`.
`oswslabinfo`	Contains a listing of the contents of `/proc/slabinfo`.
`oswtop`	Contains output from `top`.
`oswvmstat`	Contains output from `vmstat`.

OSWbb stores data in hourly archive files, which are named *`system_name_utility_name_timestamp`*`.dat`. Each entry in a file is preceded by a timestamp.

Analyzing OSWbb Archived Files

You can use the OSWbb analyzer (OSWbba) to provide information about system slowdowns, system delays, and other performance problems. You can also use OSWbba to graph data that's collected from the `iostat`, `netstat`, and `vmstat` utilities. OSWbba requires that you have Java version 1.4.2 or a later version installed on the system.

You can download a Java RPM for Linux by visiting http://www.java.com, or you can install Java by using the `dnf` command:

```
sudo dnf install java-1.8.0-jdk
```

Run OSWbba from the `oswbb` directory as follows:

```
sudo java -jar oswbba.jar -i OSWbba_archive
```

In the previous command, *OSWbba_archive* is the path of the archive directory that contains the OSWbb log files.

You can use OSWbba to display the following types of performance graph:

- Process run, wait, and block queues.
- CPU time spent running in system, user, and idle mode.
- Context switches and interrupts.
- Free memory and available swap.
- Reads each second, writes each second, service time for I/O requests, and percentage usage of bandwidth for a specified block device.

You can also use OSWbba to save the analysis to a report file, which reports instances of system slowdown, spikes in run queue length, or memory shortage, describes probable causes, and offers suggestions of how to improve performance.

```
sudo java -jar oswbba.jar -i OSWbba_archive -A
```

For more information about OSWbb and OSWbba, refer to the OSWatcher Black Box User Guide (Article ID 301137.1) and the OSWatcher Black Box Analyzer User Guide (Article ID 461053.1) on My Oracle Support (MOS) at https://support.oracle.com.

6
Working With Performance Co-Pilot

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Collecting and Analyzing Metrics With Performance Co-Pilot.

7

Automating System Tasks

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Automating System Tasks With cron.

8
Configuring and Using Auditing

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Auditing the System With Auditd and Rsyslogd.

9
Working With Kernel Dumps

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Managing Kernels and System Boot.

10

Working With Core Dumps

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Managing the System With systemd.

11

Working With the drgn and corelens Kernel Debugging Utilities

> **Note**
>
> The information in this chapter has been migrated to a separate and more updated documentation. See Oracle Linux 8: Debugging the Kernel With Drgn and Corelens.