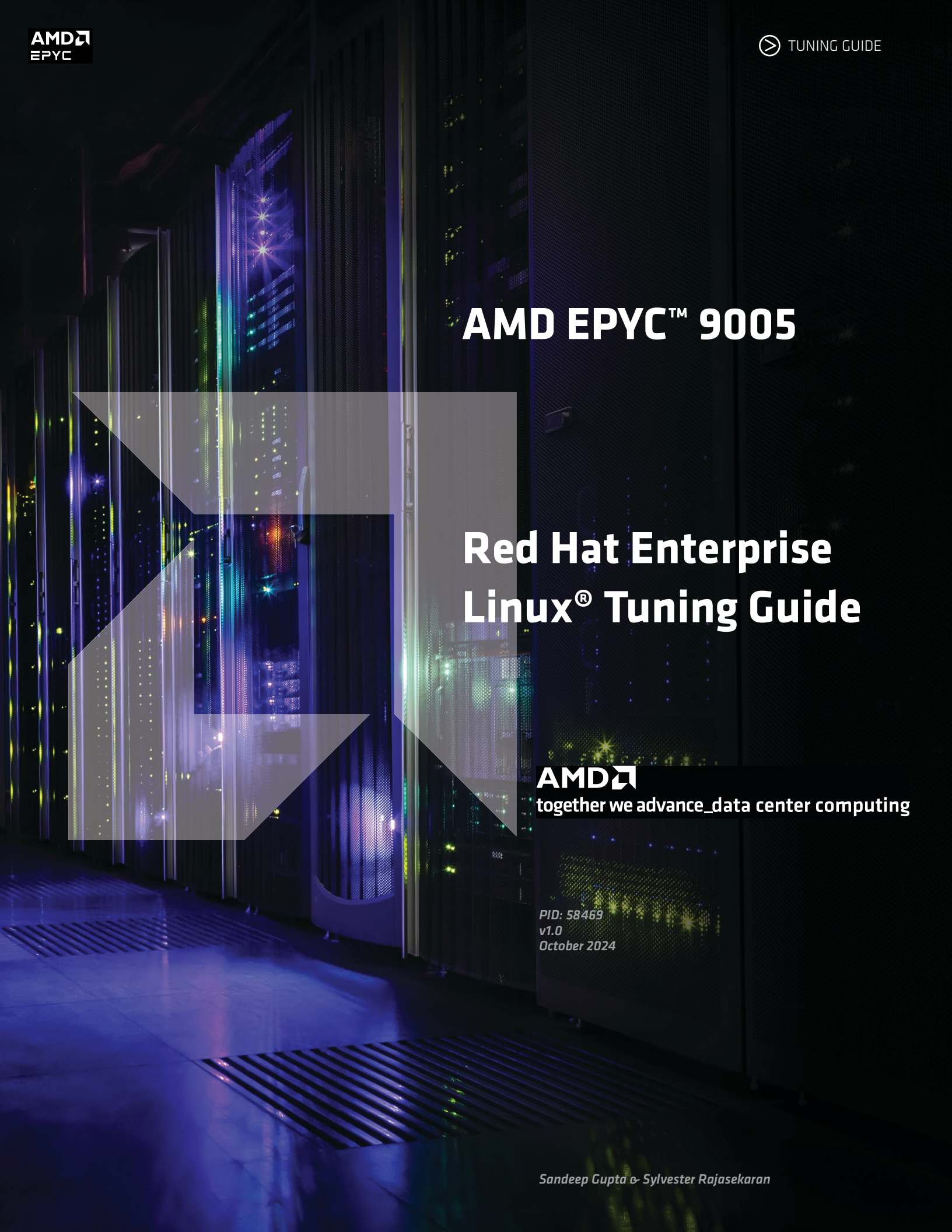

AMD EPYC

TUNING GUIDE

# AMD EPYC™ 9005

# Red Hat Enterprise Linux® Tuning Guide

AMD
together we advance_data center computing

PID: 58469
v1.0
October 2024

Sandeep Gupta & Sylvester Rajasekaran

| Date | Version | Changes |
|---|---|---|
| Jun, 2024 | 0.1 | Initial NDA release |
| October, 2024 | 1.0 | Initial public release |
| | | |
| | | |
| | | |
| | | |

# Audience

This document is intended for a technical audience such as Linux® application architects, production deployment, and performance engineering teams with a server configuration background who have:

- Admin access to the server's management interface (BMC).
- Familiarity with the server's management interface.
- Admin OS access.
- Familiarity with the OS-specific configuration, monitoring, and troubleshooting tools.

*Note: Verbose commands have been truncated for readability and are intended to show sample outputs only.*

# RED HAT ENTERPRISE LINUX®
# TUNING GUIDE

## CONTENTS

# Chapter 1: Introduction

This tuning guide describes parameters that can optimize performance of servers powered by AMD EPYC™ 9005 Series Processors running Red Hat Enterprise Linux (RHEL) Operating Systems.

Chapter 4 describes some of the available Linux tuning tools. Please also see Red Hat Enterprise Linux 9 - Monitoring and Managing System Performance* (PDF) for more information on these toolkits and RHEL 9-specific monitoring, system performance, and tuning settings. Please further visit Monitoring and managing system status and performance* (customer portal).

This Tuning Guide uses examples based on RHEL 9.5, Red Hat Enterprise Linux 9.X (Plow) running Linux 5.14.0-427.13.1.el9_4.x86_64.

## 1.1 - Networking Support

AMD tested several adapters at multiple speeds from 1Gbps to 200Gbps, using the recommendations and from the following tables in the *Linux Network Tuning Guide for AMD EPYC 9005 Series Processors* (available from the AMD Documentation Hub):

- Network Tuning Recommendations
- Network Testing Results

## 1.2 - Important Reading

Please be sure to read the following guides (available from the AMD Documentation Hub), which contain important foundational information about 5th Gen AMD EPYC processors:

- *AMD EPYC™ 9005 Processor Architecture Overview*
- *BIOS & Workload Tuning Guide for AMD EPYC™ 9005 Series Processors*
- *Memory Population Guidelines for AMD EPYC™ 9005 Series Processors*

THIS PAGE INTENTIONALLY LEFT BLANK.

# Chapter 2: Common Linux Kernel Tools and Examples

The Red Hat Enterprise Linux kernel has had NUMA-aware default memory and CPU scheduling policy since RHEL 6, and this support is being enhanced across all current versions of RHEL 9. Please visit Monitoring and managing system status and performance* (customer portal) for additional information about RedHat Enterprise Linux NUMA architecture, manual NUMA binding using `numactl`, automatic NUMA binding using `numad`, kernel automatic NUMA balancing, and more.

NUMA (Non Uniform Memory Access) systems consist of CPU clusters or CPU groups. Each CPU group is called a NUMA node, and each NUMA node has its own CPUs, memory, and I/O devices. NUMA nodes connect to memory and I/O devices on remote CPUs via one or more buses (or interconnects). The term NUMA comes from the fact that it is faster to access local memory than memory associated with other NUMA nodes.

NUMA architecture introduces memory access latencies depending on the distance between the CPU and the memory location. System BIOS populates the System Locality Information Table (SLIT) and supplies it to the Linux kernel via the Advanced Configuration and Power Interface (ACPI) and provides the normalized distances between the different NUMA nodes. See Socket SP5/SP6 Platform NUMA Topology for AMD Family 1Ah Models 00h–0Fh and Models 10h–1Fh - Login required; please review the latest version if multiple versions are present.

The NUMA tools in Red Hat Enterprise Linux improve application performance on modern hardware systems. Additional tuning can improve performance even more. For example, Red Hat created the optional user-level `numad` daemon that provides process management and placement advice for efficient CPU and memory usage on systems with NUMA topology.

Automatic NUMA balancing provides satisfactory performance in most cases. and the default performance is near optimal. This chapter lists some commonly available Linux management tools and provides some examples. See:

## 2.1 - LSCPU

`lscpu` gives a quick view of CPU topology with the following information:

- Number of sockets, nodes, cores, and threads present in the system.
- Caches and their sizes.
- NUMA nodes and CPU associations.

### 2.1.1 - Example

```
# lscpu

Architecture:            x86_64
  CPU op-mode(s):        32-bit, 64-bit
  Address sizes:         46 bits physical, 57 bits virtual
  Byte Order:            Little Endian
CPU(s):                  512
  On-line CPU(s) list:   0-511
Vendor ID:               AuthenticAMD
  BIOS Vendor ID:        Advanced Micro Devices, Inc.
  Model name:            AMD EPYC 9755 128-Core Processor
    BIOS Model name:     AMD EPYC 9755 128-Core Processor
    CPU family:          26
    Model:               2
    Thread(s) per core:  2
    Core(s) per socket:  128
    Socket(s):           2
    Stepping:            1
    Frequency boost:     enabled
    CPU(s) scaling MHz:  66%
    CPU max MHz:         4121.1909
    CPU min MHz:         1500.0000
    BogoMIPS:            5392.13
    Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush
mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm constant_tsc rep_go od amd_lbr_v2 nopl
nonstop_tsc cpuid extd_apicid aperfmperf rapl pni pclmulqdq monitor ssse3 fma cx16 pcid sse4_1 sse4_2
x2apic movbe popcnt aes xsave avx f16c rdran d lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a
misalignsse 3dnowprefetch osvw ibs skinit wdt tce topoext perfctr_core perfctr_nb bpext perfctr_llc mwaitx
cpb  cat_l3 cdp_l3 hw_pstate ssbd mba perfmon_v2 ibrs ibpb stibp ibrs_enhanced vmmcall fsgsbase tsc_adjust
bmi1 avx2 smep bmi2 erms invpcid cqm rdt_a avx512f avx512dq rds eed adx smap avx512ifma clflushopt clwb
avx512cd sha_ni avx512bw avx512vl xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc cqm_mbm_total
cqm_mbm_local avx_vnni avx512_bf16 clzero irperf xsaveerptr rdpru wbnoinvd amd_ppin cppc arat npt lbrv
svm_lock nrip_save tsc_scale vmcb_clean flushbyasid decodeassists pausefilter pfthresh old avic
v_vmsave_vmload vgif x2avic v_spec_ctrl vnmi avx512vbmi umip pku ospke avx512_vbmi2 gfni vaes vpclmulqdq
avx512_vnni avx512_bitalg avx512_vpopcntdq la57 rdpid bus_lock_detect movdiri movdir64b overflow_recov
succor smca fsrm avx512_vp2intersect flush_l1d sev sev_es debug_swap
Virtualization features:
  Virtualization:        AMD-V
Caches (sum of all):
  L1d:                   12 MiB (256 instances)
  L1i:                   8 MiB (256 instances)
  L2:                    256 MiB (256 instances)
  L3:                    1 GiB (32 instances)

NUMA:
  NUMA node(s):          2
  NUMA node0 CPU(s):     0-127,256-383
  NUMA node1 CPU(s):     128-255,384-511

Vulnerabilities:
  Gather data sampling:  Not affected
  Itlb multihit:         Not affected
  L1tf:                  Not affected
```

```
  Mds:                    Not affected
  Meltdown:               Not affected
  Mmio stale data:        Not affected
  Retbleed:               Not affected
  Spec rstack overflow:   Not affected
  Spec store bypass:      Mitigation; Speculative Store Bypass disabled via prctl
  Spectre v1:             Mitigation; usercopy/swapgs barriers and __user pointer sanitization
  Spectre v2:             Mitigation; Enhanced / Automatic IBRS, IBPB conditional, STIBP always-on, RSB
filling, PBRSB-eIBRS Not affected
  Srbds:                  Not affected
  Tsx async abort:        Not affected
```

## 2.2 - NUMACTL

`numactl` can control the NUMA policy for processes or shared memory. `numactl` and `numad` help tune NUMA scheduling parameters and monitor both NUMA topology and resource utilization by automatically making affinity adjustments to locally optimize processes. You can use `numactl` to find:

- Number of NUMA nodes in the system.
- Distance between nodes.
- CPU and memory association with the node.
- Linux policy (default, bind, preferred, interleave) of the current process.

The second part of the `numactl –hardware` output gives the node distances in a matrix based on the System Locality Information Table in the Advanced Configuration and Power Interface (ACPI SLIT). These distances indicate the cost of accessing remote memory as the relative latency to access memory from a particular node, normalized to a base value of 10. Higher values indicate more overhead.

### 2.2.1 - Example

```
# dnf install numactl


# numactl –hardware

available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 128 129 130 131 132
133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186
187 188 189 190 191
node 0 size: 31455 MB
node 0 free: 27729 MB
node 1 cpus: 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94
95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122
123 124 125 126 127 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213
214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240
241 242 243 244 245 246 247 248 249 250 251 252 253 254 255
node 1 size: 32149 MB
node 1 free: 29874 MB
node distances:
node   0   1
  0:  10  32
  1:  32  10
```

## 2.3 - LSTOPO

The graphical output consists of nested boxes representing the inclusion of objects in the resource hierarchies. A **Machine** box usually contains one or several **Package** boxes that contain multiple **Core** boxes that each have one or more PUs.

Caches appear in a slightly different manner because they do not actually include computing resources, such as cores. For instance, a L2 Cache shared by a pair of cores appears as a **Cache** box on top of two **Core** boxes (instead of having **Core** boxes inside the **Cache** box).

By default, NUMA node boxes appear on top of their local computing resources. For instance, a processor **Package** containing one NUMA node and four **Cores** appears as a **Package** box containing the NUMA node box above four **Core** boxes. If a NUMA node is local to the L3 Cache, then the NUMA node appears above that **Cache** box.

The PCI hierarchy does not appear as a set of included boxes; instead, it appears as a tree of bridges (that may actually be switches) with links between them. The tree starts with a small square on the left for the host bridge or root complex. It ends with PCI device boxes on the right. Intermediate PCI bridges/switches may appear as additional small squares in the middle.

Please see https://github.com/open-mpi/hwloc/tree/master/utils/lstopo* for additional information.

### 2.3.1 - Example

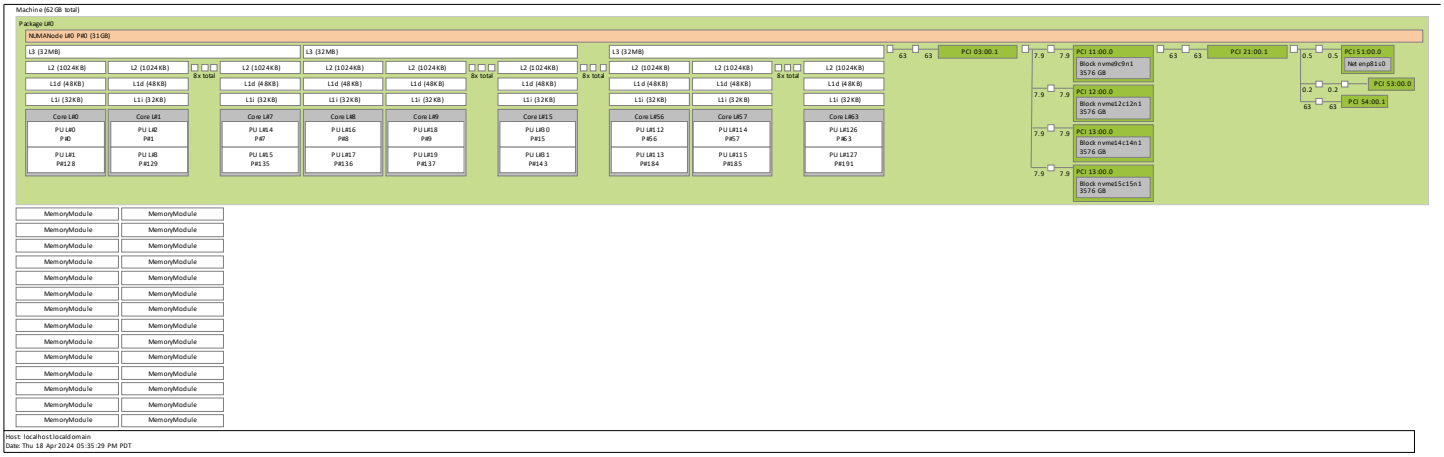

*Figure 2-1: Sample lstopo output (1 of 2)*

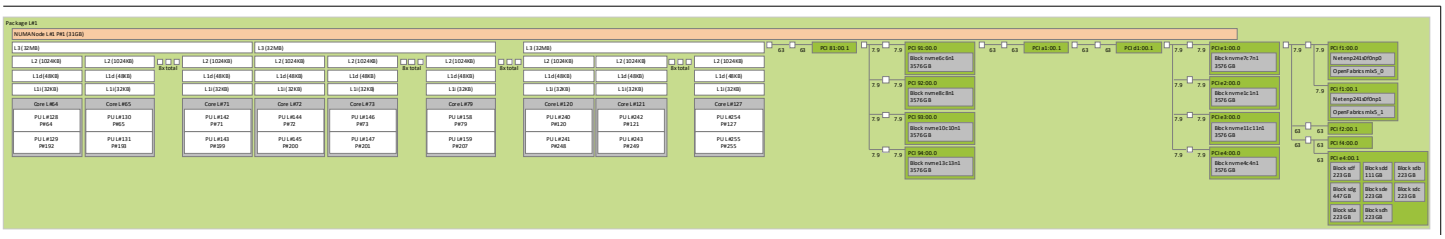

*Figure 2-2: Sample lstopo output (2 of 2)*

## 2.4 - Portable Hardware Locality (hwloc) and hwloc-gui

`hwloc` and `hwloc-gui` help you visualize the system NUMA topology. `hwloc` includes other packages that help with viewing different aspects of NUMA management.

```
# yum install hwloc
# yum install hwloc-gui
```

### 2.4.1 - hwloc-ls

`hwloc-ls` obtains CPU IDs when you need to know which cores to pin to your job to.

```
# hwloc-ls

Machine (62GB total)
  Package L#0
    NUMANode L#0 (P#0 31GB)
    L3 L#0 (32MB)
      L2 L#0 (1024KB) + L1d L#0 (48KB) + L1i L#0 (32KB) + Core L#0
        PU L#0 (P#0)
        PU L#1 (P#128)
      L2 L#1 (1024KB) + L1d L#1 (48KB) + L1i L#1 (32KB) + Core L#1
.............
.............
.................
           HostBridge
      PCIBridge
        PCI 03:00.1 (ProcessingAccelerator)
    HostBridge
      PCIBridge
        PCI 11:00.0 (NVMExp)
          Block(Disk) "nvme12c12n1"
.............
.............
.................
        PCI 21:00.1 (ProcessingAccelerator)
    HostBridge
      PCIBridge
        PCI 51:00.0 (Ethernet)
          Net "enp81s0"
.............
.............
.................
    NUMANode L#1 (P#1 31GB)
.............
.............
.................
      L2 L#66 (1024KB) + L1d L#66 (48KB) + L1i L#66 (32KB) + Core L#66
        PU L#132 (P#66)
        PU L#133 (P#194)
      L2 L#67 (1024KB) + L1d L#67 (48KB) + L1i L#67 (32KB) + Core L#67
        PU L#134 (P#67)
        PU L#135 (P#195)
.............
.............
.................
        PCI 91:00.0 (NVMExp)
          Block(Disk) "nvme10c10n1"
      PCIBridge
.............
.............
.................
  Misc(MemoryModule)
  Misc(MemoryModule)
  Misc(MemoryModule)
.............
```

#### 2.4.1.1 - hwloc-info

`hwloc-info` gets the system cache hierarchy.

```
# hwloc-info

depth 0:           1 Machine (type #0)
 depth 1:          2 Package (type #1)
  depth 2:         16 L3Cache (type #6)
   depth 3:        128 L2Cache (type #5)
    depth 4:       128 L1dCache (type #4)
     depth 5:      128 L1iCache (type #9)
      depth 6:     128 Core (type #2)
       depth 7:    256 PU (type #3)
Special depth -3:  2 NUMANode (type #13)
Special depth -4:  41 Bridge (type #14)
Special depth -5:  30 PCIDev (type #15)
Special depth -6:  29 OSDev (type #16)
Special depth -7:  28 Misc (type #17))
Special depth -7:  28 Misc (type #17)
```

## 2.5 - CPUPOWER

`cpupower` is a collection of tools that examine and tune processor power-saving features, such as checking the CPU governor. The `cpupower monitor` command samples the current CPU frequency in real time. From the Linux perspective, the CPU frequency scaling subsystem can modulating the CPU frequency based on workload feedback. Setting the governor to `performance` reduces OS frequency toggling. See CPU Performance Scaling* for additional information.

```
# cpupower frequency-info

analyzing CPU 182:
  driver: acpi-cpufreq
  CPUs which run at the same hardware frequency: 182
  CPUs which need to have their frequency coordinated by software: 182
  maximum transition latency:  Cannot determine or is not supported.
  hardware limits: 1.50 GHz - 3.72 GHz
  available frequency steps:  2.00 GHz, 1.70 GHz, 1.50 GHz
  available cpufreq governors: conservative ondemand userspace powersave performance schedutil
  current policy: frequency should be within 1.50 GHz and 2.00 GHz.
                  The governor "performance" may decide which speed to use
                  within this range.
  current CPU frequency: 2.00 GHz (asserted by call to hardware)
  boost state support:
    Supported: yes
    Active: yes
    Boost States: 0
    Total States: 3
    Pstate-P0:  28800MHz
    Pstate-P1:  16800MHz
    Pstate-P2:  8800MHz
```

Execute `cat /sys/devices/system/cpu/cpufreq/boost` to check whether boost is ON (1) or OFF (0).

```
# cat /sys/devices/system/cpu/cpufreq/boost
1
```

## 2.5.1 - cpupower monitor

Check core frequencies and idle states.

### 2.5.1.1 - Example

```
# cpupower monitor
               | Mperf              || Idle_Stats
 PKG|CORE| CPU| C0   | Cx    | Freq  || POLL | C1    | C2
   0|   0|   0|  0.35| 99.65|  2298||  0.00|  0.00| 99.64
   0|   0| 128|  0.04| 99.96|  2126||  0.00|  1.55| 98.41
   0|   1|   1|  0.03| 99.97|  1626||  0.00|  0.00| 99.97
   0|   1| 129|  0.03| 99.97|  1503||  0.00|  0.78| 99.19
   0|   2|   2|  0.04| 99.96|  1603||  0.01|  0.00| 99.97
   0|   2| 130|  0.04| 99.96|  1491||  0.00|  0.78| 99.19
   0|   3|   3|  0.03| 99.97|  1615||  0.00|  0.00| 99.97
   0|   3| 131|  0.04| 99.96|  1492||  0.00|  0.78| 99.19
   0|   4|   4|  0.04| 99.96|  1589||  0.01|  0.00| 99.96
   0|   4| 132|  0.04| 99.96|  1491||  0.00|  0.78| 99.19
   0|   5|   5|  0.11| 99.89|  1822||  0.00|  0.00| 99.89
   0|   5| 133|  0.04| 99.96|  1484||  0.00|  0.78| 99.19
   0|   6|   6|  0.17| 99.83|  2168||  0.00|  0.00| 99.84
   0|   6| 134|  0.03| 99.97|  1647||  0.00|  0.78| 99.19
   0|   7|   7|  0.04| 99.96|  1587||  0.01|  0.00| 99.96
   0|   7| 135|  0.03| 99.97|  1481||  0.00|  0.78| 99.19
   0|   8|  48|  0.04| 99.96|  1959||  0.00|  0.78| 99.19
   0|   8| 176|  0.04| 99.96|  2088||  0.00|  0.87| 99.09
   0|   9|  49|  0.04| 99.96|  1986||  0.00|  0.78| 99.19
   0|   9| 177|  0.04| 99.96|  2092||  0.00|  0.80| 99.17
   0|  10|  50|  0.04| 99.96|  1987||  0.00|  0.78| 99.19
   0|  10| 178|  0.04| 99.96|  2059||  0.00|  0.89| 99.08
   0|  11|  51|  0.04| 99.96|  1991||  0.00|  0.78| 99.19
   0|  11| 179|  0.04| 99.96|  2099||  0.00|  0.89| 99.07
   0|  12|  52|  0.03| 99.97|  1990||  0.00|  0.78| 99.19
   0|  12| 180|  0.03| 99.97|  2066||  0.00|  0.89| 99.08
.............
.............
.............
   0|  24|  32|  0.04| 99.96|  1924||  0.01|  0.00| 99.96
   0|  24| 160|  0.04| 99.96|  2045||  0.00|  0.78| 99.18
   0|  25|  33|  0.04| 99.96|  1935||  0.01|  0.00| 99.96
   0|  25| 161|  0.03| 99.97|  2017||  0.00|  0.78| 99.19
   0|  26|  34|  0.04| 99.96|  1964||  0.00|  0.78| 99.19
   0|  26| 162|  0.04| 99.96|  2049||  0.00|  0.78| 99.19
   0|  27|  35|  0.04| 99.96|  1979||  0.00|  0.87| 99.09
   0|  27| 163|  0.04| 99.96|  2046||  0.00|  0.78| 99.19
   0|  28|  36|  0.04| 99.96|  1980||  0.00|  0.80| 99.16
   0|  28| 164|  0.03| 99.97|  2069||  0.00|  0.00| 99.97
   0|  29|  37|  0.04| 99.96|  1985||  0.01|  0.02| 99.94
   0|  29| 165|  0.04| 99.96|  2053||  0.00|  0.78| 99.19
   0|  30|  38|  0.04| 99.96|  1972||  0.01|  0.01| 99.85
   0|  30| 166|  0.04| 99.96|  2050||  0.00|  0.78| 99.19
   0|  31|  39|  0.04| 99.96|  1978||  0.01|  0.01| 99.86
   0|  31| 167|  0.04| 99.96|  2050||  0.00|  0.78| 99.19
   0|  32|   8|  0.05| 99.95|  1945||  0.01|  0.00| 99.96
   0|  32| 136|  0.04| 99.96|  2024||  0.00|  0.78| 99.19
   0|  33|   9|  0.04| 99.96|  1956||  0.01|  0.00| 99.96
   0|  33| 137|  0.03| 99.97|  2028||  0.00|  0.78| 99.19
   0|  34|  10|  0.04| 99.96|  1957||  0.01|  0.00| 99.96
   0|  34| 138|  0.03| 99.97|  2023||  0.00|  0.78| 99.19
.............
.............
.............
```

## 2.6 - CPU Governors

AMD EPYC supports several CPU governors. Different governors can be applied to different cores. For example, the performance governor is often used in a High-Performance Computing environment.

- **performance:** Sets the core frequency to the highest available frequency within P0.
- **Boost=OFF:** The CPU will operate at the base frequency, e.g., 2.25GHz on an AMD EPYC 7742 CPU.
- **Boost=ON:** The CPU will attempt to boost the frequency up to the Max Boost frequency of 3.4Ghz. While operating at the boosted frequencies, this still represents the P0 P-state.
- **ondemand:** Sets the core frequency depending on the trailing load. This favors a rapid ramp to the highest operating frequency with a subsequent slow step down to P2 when idle. This could penalize short-lived threads.
- **conservative:** Similar to `ondemand` but favors a more graceful ramp to highest frequency and a rapid return to P2 at idle.
- **powersave:** Sets the lowest-supported core frequency, locking it to P2.

Administrators can execute the `cpupower` command to set the CPU governor. For example, to set the CPU governor to Performance:

```
cpupower frequency-set -g performance
```

Please see Linux CPUFreq Governors* for a more extensive discussion and explanation of Linux CPU governors.

Here are some examples of using `cpupower` to query and set a range of conditions on the CPU:

```
cpupower -c 0-15 monitor
```

Displays the frequencies of cores 0 to 15. Useful if a user needs to observe the changes while turning Boost ON and OFF.

```
cpupower frequency-info
```

Lists the boost state, CPU governor, and other useful information about the CPU configuration.

```
cpupower frequency-set -g performance
```

Changes the CPU governor to `performance`.

```
cpupower -c 0-15 idle-set -d 2
```

Disables the C2 idle state on CPUs 0 to 15.

## 2.7 - CPU Performance Scaling Drivers

CPU performance scaling drivers implement the CPU-specific frequency settings specified by the governor. The ACPI standard requires P-states that start at P0 (highest performance) and proceed through lower-performing states; however, AMD EPYC processors allow specifying specific frequencies. The applicable scaling drivers are:

- **amd_pstate:** This driver has three modes that correspond to `active`, `passive`, and `guided` degrees of autonomy from the CPU hardware and automatically loads in `active` mode on supported "Zen 2" and newer AMD EPYC processors. See amd-pstate CPU Performance Scaling Driver* for additional details.
- **acpi_cpufreq:** This driver uses the ACPI processor P-states.

## 2.8 - top

`top` provides a dynamic view of the resources being consumed by various processes. Please see "Default top Options" on page 11.

While running `top`, you can

- Press [1] to view a per-CPU breakdown of utilization statistics.
- Press [2] to view per-NUMA-node utilization statistics.
- Press [3] to select and highlight a NUMA node and view summary information. See "top - 15:23:56 up 11 days, 1:50, 4 users, load average: 0.97, 0.52, 0.30" on page 13.

### 2.8.1 - Default top Options

#### 2.8.1.1 - Example

```
top - 15:20:34 up 11 days,  1:46,  4 users,  load average: 0.15, 0.05, 0.12
Tasks: 2345 total,   2 running, 2343 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.4 us,  0.0 sy,  0.0 ni, 99.6 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem : 63604.5 total, 58557.1 free,  4838.0 used, 820.0 buff/cache
MiB Swap:   31996.0 total,   31996.0 free,      0.0 used. 58766.5 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  17895 root      20   0  264260  40960   4096 R  99.7   0.1   0:08.86 perl
  17896 root      20   0  228524   4096   2048 R   1.0   0.0   0:00.06 top
      1 root      20   0  174476  26624   8192 S   0.0   0.0   0:04.45 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.25 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 knetns
      8 root      20   0       0      0      0 I   0.0   0.0   0:00.98 kworker/0:0H-events_highpri
     12 root      20   0       0      0      0 S   0.0   0.0   0:00.00 mm_percpu_u
     13 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_kthre
     14 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_rude_
     15 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_trace
     16 root      rt   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/0
     17 root      20   0       0      0      0 S   0.0   0.0   0:00.00 pr/tty0
     18 root      20   0       0      0      0 I   0.0   0.0   4:01.05 rcu_preempt
     19 root      rt   0       0      0      0 S   0.0   0.0   0:00.49 migration/1
     20 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     22 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
     23 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
     24 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/1
```

```
     25 root      rt   0       0      0      0 S   0.0   0.0   0:00.00 migration/1
     26 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/1
     28 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 0:0H-events_highpri
     29 root      20   0       0      0      0 S   0.0   0.0   0:00.50 cpuhp/2
     30 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/2
     31 root      rt   0       0      0      0 S   0.0   0.0   0:00.00 migration/2
     32 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/2
     33 root      20   0       0      0      0 I   0.0   0.0   0:00.00 kworker/2:0-events
     34 root       0 -20       0      0      0 I   0.0   0.0   0:00.49 kworker/2:0H-events_highpri
     35 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/3
     36 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/3
     37 root      rt   0       0      0      0 S   0.0   0.0   0:00.00 migration/3
     38 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/3
     39 root      20   0       0      0      0 I   0.0   0.0   0:00.49 kworker/3:0-events
     40 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/3:0H-events_highpri
     41 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/4
.............
.............
...................
```

## 2.8.2 - Per-CPU Utilization Statistics

### 2.8.2.1 - Example

Press [1] to view a per-CPU breakdown of utilization statistics.

```
top - 15:21:19 up 11 days,  1:47,  4 users,  load average: 0.60, 0.19, 0.16
Tasks: 2345 total,   2 running, 2343 sleeping,   0 stopped,   0 zombie
%Cpu0  :  0.0 us,  0.0 sy,  0.0 ni, 99.0 id,  0.0 wa,  0.0 hi,  1.0 si,  0.0 st
%Cpu1  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu2  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu3  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu4  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu5  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu6  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu7  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu8  :  8.2 us, 20.6 sy,  0.0 ni, 71.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu9  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu10 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu11 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu12 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu13 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu14 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu15 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu16 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu17 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu18 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu19 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu20 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu21 :  0.3 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu22 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu23 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu24 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu25 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu26 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu27 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu28 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu29 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu30 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu31 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu32 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu33 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu34 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu35 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu36 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu37 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu38 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
```

```
%Cpu39 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu40 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu41 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
……..
……..
……..
```

## 2.8.3 - Per-NUMA-Node Utilization Statistics

### 2.8.3.1 - Example

Press [2] to view per-NUMA-node utilization statistics.

```
top - 15:23:56 up 11 days,  1:50,  4 users,  load average: 0.97, 0.52, 0.30
Tasks: 2346 total,   2 running, 2344 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.4 us,  0.0 sy,  0.0 ni, 99.6 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Node0 :  0.0 us,  0.3 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Node1 :  0.8 us,  0.0 sy,  0.0 ni, 99.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  63604.5 total,  58246.6 free,   5148.2 used,    820.4 buff/cache
MiB Swap:  31996.0 total,  31996.0 free,      0.0 used.  58456.3 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  17895 root      20   0  581304 359744   4096 R 100.0   0.6   3:30.52 perl
     18 root      20   0       0      0      0 I   0.3   0.0   4:01.09 rcu_preempt
  17897 root      20   0  228524   4096   2048 R   0.3   0.0   0:00.57 top
      1 root      20   0  173324  17836  10560 S   0.0   0.0   0:04.45 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.25 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_flushwq
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.05 kworker/0:1H-events_highpri
     12 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
     13 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_kthre
     14 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_rude_
     15 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_tasks_trace
     16 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/0
     17 root      20   0       0      0      0 S   0.0   0.0   0:00.00 pr/tty0
     19 root      rt   0       0      0      0 S   0.0   0.0   0:00.49 migration/0
     20 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     22 root      20   0       0      0      0 S   0.0   0.0   0:01.00 cpuhp/0
     23 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
     24 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/1
     25 root      rt   0       0      0      0 S   0.0   0.0   0:00.81 migration/1
     26 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/1
     28 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/1:0H-events_highpri
     29 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/2
     30 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/2
     31 root      rt   0       0      0      0 S   0.0   0.0   0:00.49 migration/2
     32 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/2
     33 root      20   0       0      0      0 I   0.0   0.0   0:00.00 kworker/2:0H-events
     34 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/2:0H-events_highpri
     35 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/3
     36 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/3
     37 root      rt   0       0      0      0 S   0.0   0.0   0:00.49 migration/3
     38 root      20   0       0      0      0 S   0.0   0.0   0:00.00 ksoftirqd/3
……..
……..
……..
```

## 2.8.4 - Utilization Statistics Summary

### 2.8.4.1 - Example 1: NUMA Node 0 Selected

Press [3] to select and highlight a NUMA node and view summary information.

```
top - 15:24:53 up 11 days,  1:51,  4 users,  load average: 0.71, 0.52, 0.31
Tasks: 2346 total,   2 running, 2344 sleeping,   0 stopped,   0 zombie
%Node0 :  0.1 us,  0.3 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu0  :  0.0 us,  0.0 sy,  0.0 ni, 99.3 id,  0.0 wa,  0.0 hi,  0.7 si,  0.0 st
%Cpu1  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu2  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu3  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu4  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu5  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu6  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu7  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu8  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu9  :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu10 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu11 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu12 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu13 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu14 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu15 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu16 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu17 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu18 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu19 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu20 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu21 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu22 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu23 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu24 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu25 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu26 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu27 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu28 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu29 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu30 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu31 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu32 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu33 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu34 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu35 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu36 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
…….
……..
……..
```

### 2.8.4.2 - Example 2: NUMA Node 1 Selected

Press [3] to select and highlight a NUMA node and view summary information.

```
top - 13:35:23 up 13 days, 19:25,  2 users,  load average: 1.10, 0.86, 0.54
Tasks: 3849 total,   2 running, 3847 sleeping,   0 stopped,   0 zombie
%Node1 :  0.8 us,  0.0 sy,  0.0 ni, 99.2 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu64 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu65 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu65 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu66 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu67 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu68 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu69 :  0.3 us,  0.3 sy,  0.0 ni, 99.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu70 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu71 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu72 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu73 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
```

```
%Cpu74 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu75 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu76 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu77 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu78 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu79 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu80 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu81 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu82 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu83 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu84 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu85 : 99.7 us,  0.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.3 hi,  0.0 si,  0.0 st
%Cpu86 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu87 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu88 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu89 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu90 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu91 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu92 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu93 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu94 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu95 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu96 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu97 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu98 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu99 :  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu100:  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu101:  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
%Cpu102:  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
.............
.............
...............
```

## 2.9 - TuneD

`TuneD` is a daemon that uses `udev` to monitor connected devices and statically and dynamically tunes system settings according to a selected profile. Tuned is distributed with a number of predefined profiles for common use cases such as high throughput, low latency, or powersave. You can modify the defined rules for each profile and customize how to tune a particular device. See the latest version of the *Low Latency Tuning Guide* (available from the AMD Documentation Hub) for information on tuning Linux for low latency.

## 2.10 - Tuna

`Tuna` simplifies adjusting tunable scheduler parameters such as thread priority and IRQ handlers, and can also isolate CPU cores and sockets. After installation, execute the tuna command without any arguments to start the Tuna graphical user interface (GUI). You can also execute the `tuna -h` command to view available Command Line Interface (CLI) options. See Red Hat Enterprise Linux 9 - Monitoring and Managing System Status and Performance* for complete tuned and tune instructions.

## 2.11 - dmidecode (type 17: RAM Info: DDR5 Support)

```
# dmidecode --type 17

# dmidecode 3.5
Getting SMBIOS data from sysfs.
SMBIOS 3.7.0 present.
# SMBIOS implementations newer than version 3.5.0 are not
# fully supported by this version of dmidecode.

Handle 0x0035, DMI type 17, 92 bytes
Memory Device
        Array Handle: 0x0032
        Error Information Handle: 0x0034
        Total Width: Unknown
        Data Width: Unknown
        Size: No Module Installed
        Form Factor: Unknown
        Set: None
        Locator: DIMM 0
        Bank Locator: P0 CHANNEL A
        Type: Unknown
        Type Detail: Unknown

Handle 0x0037, DMI type 17, 92 bytes
Memory Device
        Array Handle: 0x0032
        Error Information Handle: 0x0036
        Total Width: 80 bits
        Data Width: 64 bits
        Size: 32 GB
        Form Factor: DIMM
        Set: None
        Locator: DIMM 1
        Bank Locator: P0 CHANNEL A
        Type: DDR5
        Type Detail: Synchronous Registered (Buffered)
        Speed: 4800 MT/s
        Manufacturer: SK Hynix
        Serial Number: 85CC86F3
        Asset Tag: Not Specified
        Part Number: HMCG84MEBRA174N
        Rank: 1
        Configured Memory Speed: 4800 MT/s
        Minimum Voltage: 1.1 V
        Maximum Voltage: 1.1 V
        Configured Voltage: 1.1 V
        Memory Technology: DRAM
        Memory Operating Mode Capability: Volatile memory
        Firmware Version: Unknown
        Module Manufacturer ID: Bank 1, Hex 0xAD
        Module Product ID: Unknown
        Memory Subsystem Controller Manufacturer ID: Unknown
        Memory Subsystem Controller Product ID: Unknown
        Non-Volatile Size: None
        Volatile Size: 32 GB
        Cache Size: None
        Logical Size: None
.....
.....
.....
```

# Chapter 3: General Tuning Recommendations

## 3.1 - LLC as NUMA Domain

Certain datacenter applications that use a remote job scheduler to manage workloads can benefit from pinning execution to a single NUMA node and (preferably) to share a single Last-Level Cache (LLC or L3 cache) within that node. This can be done if the system BIOS includes the L3AsNumaNode setting, which creates a NUMA node for each system CCX (L3 cache) when enabled.

Enabling this setting can improve performance for highly NUMA-optimized workloads if either workloads or workload components can:

- Be pinned to cores in a CCX.
- Benefit from sharing an L3 cache.

Please see Socket SP5/SP6 Platform NUMA Topology for AMD Family 1Ah Models 00h–0Fh and Models 10h–1Fh - Login required; please review the latest version if multiple versions are present. This manual contains additional information about NUMA architecture and settings for AMD EPYC 9005 Series Processors.

| 1P System AMD EPYC 9005 Series Processor with 8 CCDs | # NUMA Nodes with LLCasNUMA is | | Memory Interleaving when LLCasNUMA is (Enabled or Disabled) |
|---|---|---|---|
| | Enabled | Disabled | |
| NPS1 | # of CCDs | 1 | Across all the channels in a socket. |
| NPS2 | # of CCDs | 2 | All the channels are divided in two groups across the channels in a group provided these channels have DIMM populated in them. |
| NPS4 | # of CCDs | 4 | All the channels are divided in four groups across the channels in a group provided these channels have DIMM populated in them. |

*Table 3-1: # of NUMA nodes and memory interleaving for a single AMD EPYC 9005 Series Processors*

| 2P System AMD EPYC 9005 Series Processor with 8 CCDs | # NUMA Nodes with LLCasNUMA is | | Memory Interleaving when LLCasNUMA is (Enabled or Disabled) |
|---|---|---|---|
| | Enabled | Disabled | |
| NPS0 | 2 x # of CCDs | 1 | Across all channels in both sockets. This is not recommended. |
| NPS1 | 2 x # of CCDs | 2 | Across all the channels in each socket. |
| NPS2 | 2 x # of CCDs | 4 | All the channels are divided in two groups in each socket. Across the channels in a group provided these channels have DIMM populated in them. |
| NPS4 | 2 x # of CCDs | 8 | All the channels are divided in four groups in each socket. Across the channels in a group provided these channels have DIMM populated in them. |

*Table 3-2: # of NUMA nodes and memory interleaving for dual AMD EPYC 9005 Series Processors*

Please see *Memory Population Guidelines for AMD EPYC 9005 Series Processors* (available from the AMD Documentation Hub) for additional information.

*Note: Not all AMD EPYC 9005 Series Processors have 8 CCDs.*

## 3.2 - AMD HSMP Interface

AMD EPYC Family 19h and later processors support Host System Management Port (HSMP) system management functionality. The HSMP is an interface that provides OS-level software with access to system management functions via a set of mailbox registers. The `amd_hsmp` driver in the `drivers/platforms/x86/ creates miscdevice /dev/hsmp` folder allows user-space programs to run `hsmp` mailbox commands.

```
$ ls -al /dev/hsmp
crw-r--r-- 1 root root 10, 123 Jan 21 21:41 /dev/hsmp
```

On the development node:

Use Write mode to run set/configure commands.

- Use Read mode to run get/status monitor commands.

Access restrictions:

- Only the root user is allowed to open the file in write mode.
- All users can open the file in read mode.

In-kernel integration:

- Other kernel subsystems can use the exported transport function `hsmp_send_message()`.
- The driver handles locking across callers.

For example, to access the hsmp device from a C program:

1. Be sure to include the headers, which define the supported messages and message IDs:
   `#include <linux/amd_hsmp.h>`

2. Open the device file, as follows:

```
int file;
file = open("/dev/hsmp", O_RDWR); if (file < 0) {
/* ERROR HANDLING; you can check errno to see what went wrong */ exit(1);
}
```

3. The following IOCTL is defined:

```
``ioctl(file, HSMP_IOCTL_CMD, struct hsmp_message *msg)``
```

   The argument is a pointer to a:

```
struct hsmp_message {
__u32 msg_id; /* Message ID */
__u16 num_args; /* Number of input argument words in message */
__u16 response_sz; /* Number of expected output/response words */
__u32 args[HSMP_MAX_MSG_LEN]; /* argument/response buffer */
__u16 sock_ind; /* socket number */
};
```

The IOCTL returns a zero on success or a non-zero on failure; you can read `errno` to see what happened.

Please see *Chapter 12: Host System Management Port (HSMP)* of the *Preliminary Processor Programming Reference (PPR) for AMD Family 1Ah Processors* for additional information (available from the AMD Developer Hub; login required).

## 3.3 - AMD uProf

AMD uProf is a performance analysis tool for applications running on Windows and Linux. Developers can use this tool to better understand application runtime performance and identify ways to optimize it. AMD uProf offers:

- **Performance Analysis:** The CPU profiling utility identifies application runtime performance bottlenecks.
- **System Analysis:** The Performance Counter Monitor utility monitors system performance metrics.
- **Power Profiling:** System-wide power profiling monitors system thermal and power characteristics.
- Energy Analysis: The Power Application Analysis utility identifies energy hotspots in Windows applications.

Please see https://developer.amd.com/amd-uprof/ and *High Performance Toolchain: Compilers, Libraries & Profilers for AMD EPYC 9005 Series Processors* (available from the AMD Documentation Hub) for more information about AMD uProf.

## 3.4 - PERF

`perf` is a powerful tool that helps monitor various OS subsystems at the server, process, or process subset level to detect and identify performance bottlenecks and possibly tune the OS. Here are two examples of perf functionality and usage:

- `perf record` samples the function calls executed by a process or processes and writes the output to perf.data.
- `perf report` reads the perf.data file and prints a human-readable report of `top` function calls grouped by function calls and ordered by call count. For example:
  - `perf record -a -e cycles sleep 30` captures 30 seconds of data for the entire system.
    ```
    # # perf record -a -e cycles sleep 30
    [ perf record: Woken up 1 times to write data ]
    [ perf record: Captured and wrote 1.833 MB perf.data (10452 samples) ]
    ```
  - `perf record -e cycles <command>` gathers profile information for a given workload.
    ```
    # perf record -e cycles ls -R > /dev/null
    [ perf record: Woken up 2 times to write data ]
    [ perf record: Captured and wrote 0.418 MB perf.data (10468 samples) ]
    ```

You can also use trace points or create probe points using either `perf` or `trace-cmd` to gather specific information on the OS. Describing these analyses is beyond the scope of this tuning guide.

Here are a few invocation examples of `perf` commands.

### 3.4.1 - perf list cpu

The `perf list cpu` command displays the symbolic events that you can select in the various `perf` commands using the `-e` option.

```
# perf list cpu

List of pre-defined events (to be used in -e or -M):

  cpu-cycles OR cycles                               [Hardware event]
  cpu-clock                                          [Software event]
  cpu-migrations OR migrations                       [Software event]

cpu:
  L1-dcache-loads OR cpu/L1-dcache-loads/
  L1-dcache-load-misses OR cpu/L1-dcache-load-misses/
  L1-dcache-prefetches OR cpu/L1-dcache-prefetches/
  L1-icache-loads OR cpu/L1-icache-loads/
  L1-icache-load-misses OR cpu/L1-icache-load-misses/
  dTLB-loads OR cpu/dTLB-loads/
  dTLB-load-misses OR cpu/dTLB-load-misses/
  iTLB-loads OR cpu/iTLB-loads/
  iTLB-load-misses OR cpu/iTLB-load-misses/
  branch-loads OR cpu/branch-loads/
  branch-load-misses OR cpu/branch-load-misses/
  branch-instructions OR cpu/branch-instructions/    [Kernel PMU event]
  branch-misses OR cpu/branch-misses/                [Kernel PMU event]
  cache-misses OR cpu/cache-misses/                  [Kernel PMU event]
  cache-references OR cpu/cache-references/          [Kernel PMU event]
  cpu-cycles OR cpu/cpu-cycles/                      [Kernel PMU event]
  instructions OR cpu/instructions/                  [Kernel PMU event]
  stalled-cycles-backend OR cpu/stalled-cycles-backend/[Kernel PMU event]
  stalled-cycles-frontend OR cpu/stalled-cycles-frontend/[Kernel PMU event]
  amd_cpu:amd_pstate_perf                            [Tracepoint event]
  cpuhp:cpuhp_enter                                  [Tracepoint event]
  cpuhp:cpuhp_exit                                   [Tracepoint event]
  cpuhp:cpuhp_multi_enter                            [Tracepoint event]
  ipi:ipi_send_cpu                                   [Tracepoint event]
```

```
  ipi:ipi_send_cpumask                              [Tracepoint event]
  kmem:mm_page_pcpu_drain                           [Tracepoint event]
  kvm:kvm_avic_kick_vcpu_slowpath                   [Tracepoint event]
  kvm:kvm_cpuid                                     [Tracepoint event]
  kvm:kvm_vcpu_wakeup                               [Tracepoint event]
  kvm:vcpu_match_mmio                               [Tracepoint event]
  percpu:percpu_alloc_percpu                        [Tracepoint event]
  percpu:percpu_alloc_percpu_fail                   [Tracepoint event]
  percpu:percpu_create_chunk                        [Tracepoint event]
  percpu:percpu_destroy_chunk                       [Tracepoint event]
  percpu:percpu_free_percpu                         [Tracepoint event]
  power:cpu_frequency                               [Tracepoint event]
  power:cpu_frequency_limits                        [Tracepoint event]
  power:cpu_idle                                    [Tracepoint event]
  power:cpu_idle_miss                               [Tracepoint event]
  syscalls:sys_enter_getcpu                         [Tracepoint event]
  syscalls:sys_exit_getcpu                          [Tracepoint event]
  xdp:xdp_cpumap_enqueue                            [Tracepoint event]
  xdp:xdp_cpumap_kthread                            [Tracepoint event]
  xen:xen_cpu_load_idt                              [Tracepoint event]
  xen:xen_cpu_set_ldt                               [Tracepoint event]
  xen:xen_cpu_write_gdt_entry                       [Tracepoint event]
  xen:xen_cpu_write_idt_entry                       [Tracepoint event]
  xen:xen_cpu_write_ldt_entry                       [Tracepoint event]
```

### 3.4.2 - perf list cache

The `perf list cache` command displays the predefined events that you can select in the various `perf` commands using the `-e` option.

```
# perf list cache

List of pre-defined events (to be used in -e or -M):


cpu:
  L1-dcache-loads OR cpu/L1-dcache-loads/
  L1-dcache-load-misses OR cpu/L1-dcache-load-misses/
  L1-dcache-prefetches OR cpu/L1-dcache-prefetches/
  L1-icache-loads OR cpu/L1-icache-loads/
  L1-icache-load-misses OR cpu/L1-icache-load-misses/
  dTLB-loads OR cpu/dTLB-loads/
  dTLB-load-misses OR cpu/dTLB-load-misses/
  iTLB-loads OR cpu/iTLB-loads/
  iTLB-load-misses OR cpu/iTLB-load-misses/
  branch-loads OR cpu/branch-loads/
  branch-load-misses OR cpu/branch-load-misses/
```

### 3.4.3 - perf stat

`perf-stat` executes a command and gathers performance counter statistics using the following syntax:

```
perf stat [-e <EVENT> | --event=EVENT] [-a] <command>perf stat [-e <EVENT> | --
event=EVENT] [-a] — <command> [<options>]perf stat [-e <EVENT> | --event=EVENT] [-a] record
[- o file] — <command> [<options>]perf stat report [-i file]
```

Here some `perf-stat` examples:

```
#perf stat -e L1-dcache-loads,L1-dcache-load-misses,L1-dcache-prefetches ls -R > /dev/null

 Performance counter stats for 'ls -R':

       506,148,806      L1-dcache-loads
         7,412,036      L1-dcache-load-misses #1.46% of all L1-dcache accesses
```

```
         2,218,484      L1-dcache-prefetches

       0.451786630 seconds time elapsed

       0.064906000 seconds user
       0.039589000 seconds sys


# perf stat -e dTLB-loads,dTLB-load-misses ls -R > /dev/null

 Performance counter stats for 'ls -R':

           430,787      dTLB-loads
             6,291      dTLB-load-misses                  #    1.46% of all dTLB cache accesses

       0.078868981 seconds time elapsed

       0.052430000 seconds user
       0.026184000 seconds sys


# perf stat -e branch-loads,branch-load-misses ls -R > /dev/null

 Performance counter stats for 'ls -R':

       195,116,774      branch-loads
         2,298,736      branch-load-misses

       0.078552034 seconds time elapsed

       0.052181000 seconds user
       0.026076000 seconds sys
```

The core AMD PMU has a 4-bit-wide per-cycle increment for each performance monitor counter. This works for most counters, but AMD EPYC Family 17h and above processors can have more than 15 events occur in a cycle. These events are called "Large Increment per Cycle" events. One example is the number of SSE/AVX FLOPs retired (event code 0x003). To count these events, two adjacent hardware PMCs have their count signals merged to form a total of 8 bits per cycle. The PERF_CTR count registers also merge so as to count up to 64 bits.

Normally, events such as instructions retired get programmed on a single counter. For example:

```
PERF_CTL0 (MSR 0xc0010200) 0x000000000053ff0c # event 0x0c, umask 0xff
PERF_CTR0 (MSR 0xc0010201) 0x0000800000000001 # r/w 48-bit count
```

The next counter at `MSRs 0xc0010202-3` either remains unused or can be used independently to count something else.

When counting Large Increment per Cycle events, such as FLOPs, we have to reserve the next counter and program the `PERF_CTL (config)` register with the Merge event `(0xFFF)`. For example:

```
PERF_CTL0 (msr 0xc0010200) 0x000000000053ff03 # FLOPs event, umask 0xff
PERF_CTR0 (msr 0xc0010201) 0x0000800000000001 # read 64-bit count, wr low 48b
PERF_CTL1 (msr 0xc0010202) 0x0000000f004000ff # Merge event, enable bit
PERF_CTR1 (msr 0xc0010203) 0x0000000000000000 # write higher 16-bits of count
```

The count widens from the normal 48 bits to 64 bits by having the second counter carry the higher 16 bits of the count in the lower 16 bits of its counter register. This version does not support mixed 48-bit and 64-bit counts. Here is an example for an AMD EPYC 9005 processor:

```
# perf stat -e cpu/instructions/,cpu/event=0x03,umask=0xff/ ls -R > /dev/null


 Performance counter stats for 'ls -R':


    18,631,756,571 cpu/instructions/
           432,477 cpu/event=0x03,umask=0xff/
```

```
      2.293402649 seconds time elapsed
      0.998640000 seconds user
      1.269751000 seconds sys
```

## 3.5 - Additional perf Examples

```
# perf record --raw-samples -c 1000001 -e ibs_op//pp -a sleep 1

[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 1.113 MB perf.data (16 samples) ]


#perf report --dump-raw-trace

# To display the perf.data header info, please use --header/--header-only options.
#

0x1198@perf.data [0x38]: event: 79
.
. ... raw event: size 56 bytes
.  0000:  4f 00 00 00 00 00 38 00 1f 00 00 00 00 00 00 00  O.....8.........
.  0010:  e8 9c 18 40 00 00 00 00 be 91 2c 7f e9 ff ff ff  ...@......,.....
.  0020:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
.  0030:  01 00 00 00 00 00 00 00                          ........

0 0 0x1198 [0x38]: PERF_RECORD_TIME_CONV
... Time Shift      31
... Time Muliplier  1075354856
... Time Zero       18446743977058931134
... Time Cycles     0
... Time Mask       0
... Cap Time Zero   1
... Cap Time Short  0
: unhandled!

0x11d0@perf.data [0x4010]: event: 69
.
. ... raw event: size 16400 bytes
.  0000:  45 00 00 00 00 00 10 40 00 02 00 00 00 00 00 00  E......@........
.  0010:  77 0e 00 00 00 00 00 00 00 00 00 00 00 00 00 00  w...............
.  0020:  00 00 00 00 00 00 00 00 ff ff ff ff ff ff ff ff  ................
.  0030:  78 0e 00 00 00 00 00 00 01 00 00 00 00 00 00 00  x...............
.  0040:  01 00 00 00 00 00 00 00 ff ff ff ff ff ff ff ff  ................
.  0050:  79 0e 00 00 00 00 00 00 02 00 00 00 00 00 00 00  y...............
.  0060:  02 00 00 00 00 00 00 00 ff ff ff ff ff ff ff ff  ................

…..
……..
……..
0 0 0x5360 [0xa8]: PERF_RECORD_MMAP -1/0: [0xffffffffc06e0000(0x4000) @ 0]: x /lib/modules/5.14.0-
427.1.1.el9_4.x86_64/kernel/drivers/video/fbdev/core/sysfillrect.ko.xz

0x5408@perf.data [0xa8]: event: 1
.
. ... raw event: size 168 bytes
.  0000:  01 00 00 00 01 00 a8 00 ff ff ff ff 00 00 00 00  ................
.  0010:  00 70 6e c0 ff ff ff ff 00 40 00 00 00 00 00 00  .pn......@......
.  0020:  00 00 00 00 00 00 00 00 2f 6c 69 62 2f 6d 6f 64  ......../lib/mod
.  0030:  75 6c 65 73 2f 35 2e 31 34 2e 30 2d 34 32 37 2e  ules/5.14.0-427.
.  0040:  31 2e 31 2e 65 6c 39 5f 34 2e 78 38 36 5f 36 34  1.1.el9_4.x86_64
.  0050:  2f 6b 65 72 6e 65 6c 2f 64 72 69 76 65 72 73 2f  /kernel/drivers/
.  0060:  76 69 64 65 6f 2f 66 62 64 65 76 2f 63 6f 72 65  video/fbdev/core
.  0070:  2f 73 79 73 63 6f 70 79 61 72 65 61 2e 6b 6f 2e  /syscopyarea.ko.
.  0080:  78 7a 00 00 00 00 00 00 00 00 00 00 00 00 00 00  xz..............
.  0090:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

```
…….
……..
……..
# perf record -vv --raw-samples -c 100001 -e ibs_op/cnt_ctl=1/pp -a -C 0 taskset -c 0 true

Using CPUID AuthenticAMD-26-2-0
Attempt to add: ibs_op/cnt_ctl=0x1/
..after resolving event: ibs_op/cnt_ctl=0x1/
DEBUGINFOD_URLS=
…….
……..
……..
sys_perf_event_open: pid -1  cpu 0  group_fd -1  flags 0 = 7
mmap size 528384B
0x56250b7b5f38: mmap mask[0]:
Synthesizing TSC conversion information
Synthesizing id index
[ perf record: Woken up 1 times to write data ]
Looking at the vmlinux_path (8 entries long)
Using /proc/kcore for kernel data
Using /proc/kallsyms for symbols
Writing cache: %sdt_rtld:unmap_start=unmap_start
Cache committed: 0
Writing cache: %sdt_rtld:unmap_complete=unmap_complete
Cache committed: 0
Writing cache: %sdt_rtld:map_start=map_start
Cache committed: 0
Writing cache: %sdt_rtld:reloc_start=reloc_start
Cache committed: 0
Writing cache: %sdt_rtld:map_complete=map_complete
Cache committed: 0
Writing cache: %sdt_rtld:reloc_complete=reloc_complete
Cache committed: 0
Writing cache: %sdt_rtld:init_start=init_start
Cache committed: 0
Writing cache: %sdt_rtld:init_complete=init_complete
Cache committed: 0
Writing cache: %sdt_rtld:lll_lock_wait_private=lll_lock_wait_private
Cache committed: 0
Writing cache: %sdt_rtld:lll_lock_wait=lll_lock_wait
Cache committed: 0
Writing cache: %sdt_rtld:setjmp=setjmp
Cache committed: 0
Writing cache: %sdt_rtld:longjmp=longjmp
Cache committed: 0
Writing cache: %sdt_rtld:longjmp_target=longjmp_target
Cache committed: 0
failed to write feature HYBRID_TOPOLOGY
[ perf record: Captured and wrote 1.094 MB perf.data (20 samples) ]

# perf mem record -a sleep 5

[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 4.499 MB perf.data (2382 samples) ]


# perf mem report
```

```
Samples: 2K of event 'ibs_op//', Event count (approx.): 4615
Overhead       Samples  Local Weight  Memory access                        Symbol                                  Shared Object          Data Symbol
  13.02%             1  601           Remote node, same socket Any cache hit [k] timekeeping_advance                 [kernel.kallsyms]      [k] shadow_timekeeper+0
  12.76%           589  0             N/A                                    [k] perf_ibs_handle_irq                 [kernel.kallsyms]      [k] 0000000000000000
   5.55%           256  0             N/A                                    [k] perf_ibs_start                      [kernel.kallsyms]      [k] 0000000000000000
   4.44%             1  205           L3 hit                                 [.] sched_setaffinity@plt               perf                   [.] 0x000055ade80e6168
   3.12%           144  0             N/A                                    [k] nmi_restore                         [kernel.kallsyms]      [k] 0000000000000000
   2.80%             3  43            L3 hit                                 [k] io_idle                             [kernel.kallsyms]      [k] boot_cpu_data+0x38
   2.67%             3  41            L3 hit                                 [k] perf_adjust_freq_unthr_context      [kernel.kallsyms]      [k] 0xff243a63e1362548
   1.89%             1  87            L3 hit                                 [.] sched_setaffinity@plt               perf                   [.] 0x000055ade80e6168
   1.39%            64  0             N/A                                    [k] native_sched_clock                  [kernel.kallsyms]      [k] 0000000000000000
   1.19%             1  55            L3 hit                                 [k] __sysvec_call_function              [kernel.kallsyms]      [k] apic+0x0
   1.19%             1  55            L3 hit                                 [k] psi_account_irqtime                 [kernel.kallsyms]      [k] 0xff243a6a1382c6d0
   1.17%             1  54            RAM hit                                [k] mem_cgroup_css_rstat_flush          [kernel.kallsyms]      [k] 0xff243a69ef7ba9c0
   1.15%             1  53            L3 hit                                 [k] __sysvec_call_function              [kernel.kallsyms]      [k] apic+0x0
   1.15%             1  53            L3 hit                                 [k] apic_ack_irq                        [kernel.kallsyms]      [k] apic+0x0
```

```
# perf mem -t load report --sort=mem -stdio

# To display the perf.data header info, please use --header/--header-only options.
#
#
# Total Lost Samples: 0
#
# Samples: 2K of event 'ibs_op//'
# Total weight : 8564
# Sort order   : mem
#
# Overhead       Samples  Memory access
# ........  ............  ....................................
#
    28.74%             1  I/O hit
    21.22%          1817  N/A
    13.92%            24  L3 hit
    12.46%             2  Remote core, same node Any cache hit
    11.56%             1  Remote node, same socket Any cache hit
     6.05%           518  L1 hit
     4.33%             2  RAM hit
     1.72%            16  L2 hit
     0.01%             1  LFB/MAB hit


#
# (Tip: Order by the overhead of source file name and line number: perf report -s srcline)
#

# perf c2c record -a sleep 10

[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 4.655 MB perf.data (3986 samples) ]


# perf c2c report
```

| | Cacheline | | | Tot | Load Hitm | | | Total | Total | Total | Stores | | | Core Load Hit | | | LLC Load Hit | | RMT Load | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Index | Address | Node | PA cnt | Hitm | Total | LclHitm | RmtHitm | records | Loads | Stores | L1Hit | L1Miss | N/A | FB | L1 | L2 | LclHit | LclHitm | RmtHit | R |
| 0 | 0xff243a62c3e32380 | 0 | 3 | 4.35% | 1 | 0 | 1 | 3 | 3 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | |
| 1 | 0xff243a639f1cf9c0 | 0 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 2 | 0xff243a639f1cfac0 | 0 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 3 | 0xff243a69f00f2680 | 0 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 4 | 0xff243a6a13a80000 | 0 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 5 | 0xff243a6a15e5b380 | 0 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 6 | 0xff243a6a16a85800 | 0 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 7 | 0xff243a6a16a99000 | 0 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 8 | 0xff243a6aaeaa6580 | 1 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |
| 9 | 0xff243a6aaeaa6680 | 1 | 1 | 4.35% | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | |

```
# perf record -a -e ibs_fetch//

[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 1.404 MB perf.data (6021 samples) ]


#perf report -D --stdio --header


# ========
# captured on    : Thu May  9 17:27:00 2024
# header version : 1
# data offset    : 4504
# data size      : 1467456
# feat offset    : 1471960
# hostname : localhost.localdomain
# os release : 5.14.0-427.1.1.el9_4.x86_64
# perf version : 5.14.0-427.el9.x86_64
# arch : x86_64
# nrcpus online : 256
# nrcpus avail : 256
# cpudesc : AMD Eng Sample: 100-000001537-03
# cpuid : AuthenticAMD,26,2,0
# total memory : 65131020 kB
# cmdline : /usr/bin/perf record -a -e ibs_fetch//
# event : name = ibs_fetch//, , id = { 8361, 8362, 8363, 8364, 8365, 8366, 8367, 8368, 8369, 8370, 8371,
8372, 8373, 8374, 8375, 8376, 8377, 8378, 8379, 8380, 8381, 8382, 8383, 8384, 8385, >
# event : name = dummy:HG, , id = { 8617, 8618, 8619, 8620, 8621, 8622, 8623, 8624, 8625, 8626, 8627, 8628,
8629, 8630, 8631, 8632, 8633, 8634, 8635, 8636, 8637, 8638, 8639, 8640, 8641, 864>
# CPU_TOPOLOGY info available, use -I to display
# NUMA_TOPOLOGY info available, use -I to display
# pmu mappings: cpu = 4, amd_df = 12, amd_iommu_0 = 16, amd_iommu_1 = 17, amd_iommu_2 = 18, amd_iommu_3 =
19, amd_iommu_4 = 20, amd_iommu_5 = 21, amd_iommu_6 = 22, amd_iommu_7 = 23, amd_l3 >
# CACHE info available, use -I to display
# time of first sample : 14878.316883
# time of last sample : 14949.962221
# sample duration :  71645.338 ms
# MEM_TOPOLOGY info available, use -I to display
# bpf_prog_info 2: bpf_prog_7cc47bbf07148bfe_hid_tail_call addr 0xffffffffc026d970 size 105
# bpf_prog_info 21: bpf_prog_40ddf486530245f5_sd_devices addr 0xffffffffc026d788 size 310
# bpf_prog_info 22: bpf_prog_6deef7357e7b4530_sd_fw_egress addr 0xffffffffc026da10 size 55
…….
……..
……..
# bpf_prog_info 31: bpf_prog_6deef7357e7b4530_sd_fw_egress addr 0xffffffffc0274664 size 55
# bpf_prog_info 32: bpf_prog_6deef7357e7b4530_sd_fw_ingress addr 0xffffffffc02746cc size 55
# btf info of id 2
# cpu pmu capabilities: branches=16, max_precise=0
# ibs_fetch pmu capabilities: zen4_ibs_extensions=1
# ibs_op pmu capabilities: zen4_ibs_extensions=1
# missing features: TRACING_DATA BRANCH_STACK GROUP_DESC AUXTRACE STAT CLOCKID DIR_FORMAT COMPRESSED
CLOCK_DATA HYBRID_TOPOLOGY
# ========
#

0x1198@perf.data [0x38]: event: 79
.
. ... raw event: size 56 bytes
.  0000:  4f 00 00 00 00 00 38 00 1f 00 00 00 00 00 00 00  O.....8.........
.  0010:  e8 9c 18 40 00 00 00 00 be 91 2c 7f e9 ff ff ff  ...@......,.....
.  0020:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
.  0030:  01 00 00 00 00 00 00 00                          ........
…….
……..
……..
# perf mem record -- -c 10000

[ perf record: Woken up 12 times to write data ]
[ perf record: Captured and wrote 10.456 MB perf.data (88981 samples) ]
```

```
# perf mem report -F mem,sample,snoop


Samples: 88K of event 'ibs_op//', Event count (approx.): 589910
Memory access                                Samples  Snoop
N/A                                            52301  N/A
L1 hit                                         33047  N/A
LFB/MAB hit                                      131  N/A
L2 hit                                          1086  N/A
L3 hit                                          1200  N/A
L3 hit                                           682  HitM
RAM hit                                           51  N/A
Remote node, same socket RAM hit                  34  N/A
Remote core, same node Any cache hit             132  HitM
Remote core, same node Any cache hit              39  N/A
Remote node, same socket Any cache hit           133  HitM
Remote node, same socket Any cache hit            78  N/A
I/O hit                                           31  N/A
Uncached hit                                      36  N/A
```

THIS PAGE INTENTIONALLY LEFT BLANK.

# Chapter 4: Virtualization

RHEL includes virtualization functionality that allows a machine running RHEL to host multiple virtual machines (VMs), also referred to as guests. VMs use the host's physical hardware and computing resources to run a separate, virtualized operating system (guest OS) as a user-space process on the host OS.

## 4.1 - Secure Encrypted Virtualization (SEV)

AMD Secure Encrypted Virtualization (SEV) protects the data in DRAM while in use by a running VM instance. SEV encrypts the memory of each instance with a unique key. Executing the following command tells you whether the deployment is SEV-capable:

```
# lscpu | grep -i sev

Flags:                fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush mmx fxsr sse sse2 ht syscall nx mmxext fxsr_opt pdpe1gb rdtscp lm
constant_tsc rep_good nopl nonstop_tsc cpuid extd_apicid aperfmperf pni pclmulqdq
monitor ssse3 fma cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx f16c
rdrand lahf_lm cmp_legacy svm extapic cr8_legacy abm sse4a misalignsse 3dnowprefetch
osvw ibs skinit wdt tce topoext perfctr_core perfctr_nb bpext perfctr_llc mwaitx cpb
cat_l3 cdp_l3 invpcid_single hw_pstate sme ssbd mba sev ibrs ibpb stibp vmmcall
fsgsbase bmi1 avx2 smep bmi2 erms invpcid cqm rdt_a rdseed adx smap clflushopt clwb
sha_ni xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc cqm_mbm_total
cqm_mbm_local clzero irperf xsaveerptr wbnoinvd amd_ppin arat npt lbrv svm_lock
nrip_save tsc_scale vmcb_clean flushbyasid decodeassists pausefilter pfthreshold
v_vmsave_vmload vgif umip pku ospke vaes vpclmulqdq rdpid overflow_recov succor smca
```

If enabled on a virtual machine (VM), then SEV encrypts VM memory, which prevents the host from accessing data on the VM. This increases VM security if the host is successfully breached. The host hardware version determines how many VMs can use this feature simultaneously on a single host.

```
# dmesg | grep -i sev
   [    0.938380] AMD Memory Encryption Features active: SEV SEV-ES
```

Enabling SEV requires all DMA operations inside the guest to use shared memory. SEV makes this transparent to the guest by using the SWIOTLB Linux kernel pool, which has a default size of 64MB. A guest panic will occur if the Linux kernel exhausts the SWIOTLB pool. The number of devices used by the guest and the utilization of these devices directly impacts the amount of SWIOTLB required. AMD recommends increasing the SWIOTLB pool that the Linux kernel allocates for SEV guests, with 512MB as the recommended starting size.

### 4.1.1 - SEV Prerequisites

To ensure SEV support,

- The deployment must include a compute node that runs on SEV-capable AMD hardware, such as an AMD EPYC CPU.
- The deployment must include `libvirt` 4.5 or later, which includes SEV support.
- The operating system running in an encrypted instance must support SEV. For example:

```
# dmesg | grep -i -e ccp -e sev

[    1.741905] ccp 0000:54:00.5: enabling device (0000 -> 0002)
[    1.742332] ccp 0000:54:00.5: sev enabled
[    1.742334] ccp 0000:54:00.5: psp enabled
[    1.742495] ccp 0000:d1:00.5: enabling device (0000 -> 0002)
[    1.743463] ccp 0000:d1:00.5: sev enabled
[    1.743466] ccp 0000:d1:00.5: psp enabled
[    2.297362] ccp 0000:54:00.5: SEV API:1.55 build:30
[    7.418365] kvm_amd: SEV enabled (ASIDs 100 - 1006)
[    7.418367] kvm_amd: SEV-ES enabled (ASIDs 1 - 99)
```

Please see the AMD Secure Encrypted Virtualization (SEV) documentation for more information.

## 4.2 - AMD EPYC Virtualization Support

By default, the virtualization packages are not installed, as shown below. Be sure to install the virtualization packages.

```
# yum install libvirt
```

Validate that the virtualization host and packages are installed by executing the command `virt-host-validate`, and then verifying that all of the validations show PASS. If not, then adjust the required parameters as recommended in the `virt-host-validate` output.

```
# virt-host-validate

 QEMU: Checking for hardware virtualization                                 : PASS
 QEMU: Checking if device '/dev/kvm' exists                                   : PASS
 QEMU: Checking if device '/dev/kvm' is accessible                            : PASS
 QEMU: Checking if device '/dev/vhost-net' exists                             : PASS
 QEMU: Checking if device '/dev/net/tun' exists                               : PASS
 QEMU: Checking for cgroup 'cpu' controller support                           : PASS
 QEMU: Checking for cgroup 'cpuacct' controller support                       : PASS
 QEMU: Checking for cgroup 'cpuset' controller support                        : PASS
 QEMU: Checking for cgroup 'memory' controller support                        : PASS
 QEMU: Checking for cgroup 'devices' controller support                       : PASS
 QEMU: Checking for cgroup 'blkio' controller support                         : PASS
 QEMU: Checking for device assignment IOMMU support                           : PASS
 QEMU: Checking if IOMMU is enabled by kernel                                 : PASS
 QEMU: Checking for secure guest support                                      : PASS
 QEMU: Checking for AMD Secure Encrypted Virtualization-Encrypted State (SEV-ES): PASS
```

The `qemu-kvm` command allows you to view and validate AMD EPYC processor support.

```
#/usr/libexec/qemu-kvm -cpu help | grep -i epyc

x86 EPYC                  (alias configured by machine type)
x86 EPYC-Genoa            (alias configured by machine type)
x86 EPYC-Genoa-v1         AMD EPYC-Genoa Processor
x86 EPYC-IBPB             (alias of EPYC-v2)
x86 EPYC-Milan            (alias configured by machine type)
x86 EPYC-Milan-v1         AMD EPYC-Milan Processor
x86 EPYC-Milan-v2         AMD EPYC-Milan-v2 Processor
x86 EPYC-Rome             (alias configured by machine type)
x86 EPYC-Rome-v1          AMD EPYC-Rome Processor
x86 EPYC-Rome-v2          AMD EPYC-Rome Processor
x86 EPYC-Rome-v3          AMD EPYC-Rome-v3 Processor
x86 EPYC-Rome-v4          AMD EPYC-Rome-v4 Processor (no XSAVES)
x86 EPYC-v1               AMD EPYC Processor
x86 EPYC-v2               AMD EPYC Processor (with IBPB)
x86 EPYC-v3               AMD EPYC Processor
x86 EPYC-v4               AMD EPYC-v4 Processor
```

## 4.3 - Secure Nested Paging (SNP)

Secure Nested Paging (SNP) is an enhancement to Secure Encrypted Virtualization (SEV) that enables stronger memory protection. SNP is specifically designed to protect a guest VM from a malicious hypervisor by providing hardware guarantees that the hypervisor cannot use its page tables, or nested page tables, to manipulate a guest VM, such as by:

- Re-mapping guest memory to different DRAM without guest's knowledge.
- Mapping two different GPAs to the same DRAM page for the same guest.
- Writing to guest memory, causing corruption.
- Rolling back guest memory to an earlier state.

SNP includes a Reverse Map Table (RMP) data structure that only allows software manipulation of RMP entries via the new RMPUPDATE instruction. The RMP entry indicates the ownership of a DRAM page:

- **Hypervisor-owned (default):** Page can be written by the hypervisor or non-SNP guests.
- **Guest-owned:** Page can only be written by a specific guest VM
- **Hardware-owned (immutable):** Page cannot be written by any software.

The CPU tablewalker checks the RMP on every tablewalk.

The following new instructions (microcode) address SNP:

- **RMPUPDATE:**
  - Enables the hypervisor to create/modify/delete RMP entries.
  - Ensures no overlap between 4k/2MB pages.
- **PVALIDATE:** Enables the guest to "validate" a page of memory and sets RMP-validated bit.
- **PSMASH:** Allows the hypervisor to smash a 2MB page into 4k pages to enable finer-grain page controls.
- **RMPADJUST:** Allows a guest to adjust page VMPL permissions.

SNP includes the following Virtual Machine Privilege Levels (VMPL):

- Support for up to 4 VMPLs, where VMPL0 is the most privileged.
- New 8-bit field in each RMP entry for page permissions (64 bits total).
- New RMPADJUST instruction to modify permissions.
- VMPL enables a privilege hierarchy within a guest. For example:
  - VMPL0 can restrict which pages VMPL1 code can access.
  - VMPL1 can restrict which pages VMPL2 code can access, etc.
- All page are initially available only to VMPL0.

```
<HOST> #  uname -a

Linux localhost.localdomain 5.14.0-503.el9.x86_64 #1 SMP PREEMPT_DYNAMIC Thu Aug 22 14:13:39 EDT 2024
x86_64 x86_64 x86_64 GNU/Linux


<HOST> # dmesg | grep -i -e ccp -e rmp -e sev -e snp

[    0.000000] SEV-SNP: RMP table physical range [0x0000018276e00000 - 0x00000183faeffffff]
[    0.004745] SEV-SNP: Reserving start/end of RMP table on a 2MB boundary [0x00000183fae00000]
[    1.161260] AMD-Vi: IOMMU SNP support enabled.
[    1.254989] AMD-Vi: Extended features (0xa5bf732fa2295afe, 0x53f): PPR X2APIC NX GT [5] IA GA PC
GA_vAPIC SNP
[    2.806415] ccp 0000:55:00.5: enabling device (0000 -> 0002)
[    2.807781] ccp 0000:55:00.5: sev enabled
[    2.807783] ccp 0000:55:00.5: psp enabled
[    2.808116] ccp 0000:d1:00.5: enabling device (0000 -> 0002)
[    2.808987] ccp 0000:d1:00.5: sev enabled
[    2.808989] ccp 0000:d1:00.5: psp enabled
[    9.033178] ccp 0000:55:00.5: SEV API:1.55 build:37
[    9.033192] ccp 0000:55:00.5: SEV-SNP API:1.55 build:37
[   11.277328] kvm_amd: SEV enabled (ASIDs 100 - 1006)
[   11.277329] kvm_amd: SEV-ES enabled (ASIDs 1 - 99)
[   11.277330] kvm_amd: SEV-SNP enabled (ASIDs 1 - 99)
```

To enable SNP in BIOS:

1. Navigate to **Advanced > AMD CBS CPU Common Options**, then set **SMEE**, **SEV Control**, and **SNP Memory (RMP Table) Coverage** to **Enabled**.
2. Navigate to **Advanced > AMD CBS > NBIO Common Options**, then set **SEV-SNP Support** to **Enabled**.

*Note:*
`qemu-kvm` *version 9.1 has full SEV-SNP support.*
`libvirtd` (`libvirt`) *version 10.5.0 has full SEV-SNP support.*

## 4.4 - Resource Allocation and Host/VM Tuning

Please see the Red Hat Enterprise Linux documentation at Red Hat Enterprise Linux 9 - Configuring and Managing Virtualization* (PDF) for instructions on setting up a virtualization host, creating and administering VMs, and understanding the virtualization features in Red Hat Enterprise Linux

## 4.5 - Tuning the Virtualization Host

Please see the following resources for suggested I/O schedulers to improve disk performance when using Red Hat Enterprise Linux with virtualization:

- What is the suggested I/O scheduler to improve disk performance when using Red Hat Enterprise Linux with virtualization?*
- Red Hat Enterprise Linux 9 - Configuring and managing virtualization*
- Red Hat Enterprise Linux 9 - Managing storage devices*

## 4.6 - Evaluating Workloads and VM Workloads

NUMA-aware or highly parallelizable workloads can take maximum advantage of AMD EPYC 9005 Series Processor architecture for performance tuning and gains. Memory-bound, NUMA friendly workloads can be parallelized to have each thread run independently. I/O-bound workloads can support multiple devices such that each device can remain connected to the original task owner. For example:

- The original STREAM benchmark can be parallelized with OpenMP and extended to measure NUMA- aware workload performance.
- The OpenMP and MPI Versions of the NASA Parallel Benchmarks (NPB) can be another good example test workload.

Both examples make very effective test VM workloads.

Please see the *BIOS & Workload Tuning Guide for AMD EPYC™ 9005 Series Processors* available from the AMD Documentation Hub additional information.

## 4.7 - Linux Containers and Kubernetes: Red Hat OCP (Open Shift)

Please see the following resources for information on running containers on RHEL:

- Available from the AMD Documentation Hub:
  - *Kubernetes® Containers Tuning Guide for AMD EPYC™ 9005 Series Processors*
  - *OpenShift® Containers Tuning Guide for AMD EPYC™ 9005 Series Processors*
- Configuring PCI passthrough*
- OpenShift Container Platform 4.15 Documentation*

THIS PAGE INTENTIONALLY LEFT BLANK.

# Chapter 5: Troubleshooting and Debugging Notes

This section describes the following troubleshooting and debugging tools and procedures:



## 5.1 - Error Detection and Correction (EDAC)

Red Hat Enterprise Linux for AMD EPYC CPUs includes two AMD-compatible Error Detection and Correction (EDAC) modules for diagnosing memory errors:

Linux has two EDAC modules:

- `amd64_edac_mod` provides information and DRAM ECC-specific decoding. It loads automatically on supported systems if not blacklisted/disabled.
- `edac_mce_amd` provides more detailed MCA error decoding and is loaded by amd64_edac_mod.

```
# lsmod | grep -i edac
amd64_edac             49152  0
edac_mce_amd           45056  1 amd64_edac
```

System administrators monitoring ECC should monitor system health by continually recording both the number and rate of change of correctable and uncorrectable errors. The EDAC driver provides details about the number of memory controllers, memory controller characteristics, and physical DIMM characteristics (number of chip-select rows [`csrows`], and the channel tables).

The `/sys` filesystem for each `csrow` contain a number of entries with detailed information about the specific DIMM:

```
# ls -al /sys/devices/system/edac/mc/mc0/

drwxr-xr-x. 5 root root    0 May 20 14:53 .
drwxr-xr-x. 5 root root    0 May 20 14:53 ..
-r--r--r--. 1 root root 4096 May 20 14:53 ce_count
-r--r--r--. 1 root root 4096 May 20 14:53 ce_noinfo_count
drwxr-xr-x. 3 root root    0 May 20 14:53 csrow2
-r--r--r--. 1 root root 4096 May 20 14:53 max_location
-r--r--r--. 1 root root 4096 May 20 14:53 mc_name
drwxr-xr-x. 2 root root    0 May 20 14:53 power
drwxr-xr-x. 3 root root    0 May 20 14:53 rank27
--w-------. 1 root root 4096 May 20 14:53 reset_counters
-r--r--r--. 1 root root 4096 May 20 14:53 seconds_since_reset
-r--r--r--. 1 root root 4096 May 20 14:53 size_mb
-r--r--r--. 1 root root 4096 May 20 14:53 ue_count
-r--r--r--. 1 root root 4096 May 20 14:53 ue_noinfo_count
-rw-r--r--. 1 root root 4096 May 20 14:53 uevent
```

### 5.1.1 - Get the Memory Controller MCx Device Information

EDAC driver messages help identify the DIMMs.

```
# dmesg | grep -i edac

[    0.504945] EDAC MC: Ver: 3.0.0
[    7.154928] EDAC MC0: Giving out device to module amd64_edac controller F1Ah: DEV 0000:00:18.3
(INTERRUPT)
[    7.154932] EDAC amd64: F1Ah detected (node 0).
[    7.154936] EDAC MC: UMC0 chip selects:
[    7.154936] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154937] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154940] EDAC MC: UMC1 chip selects:
[    7.154941] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154941] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154944] EDAC MC: UMC2 chip selects:
[    7.154945] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154945] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154948] EDAC MC: UMC3 chip selects:
[    7.154948] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154949] EDAC amd64: MC: 2: 32768MB 3:     0MB
[    7.154952] EDAC MC: UMC4 chip selects:
[    7.154952] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154953] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154955] EDAC MC: UMC5 chip selects:
[    7.154956] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154956] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154959] EDAC MC: UMC6 chip selects:
[    7.154959] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154960] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154963] EDAC MC: UMC7 chip selects:
[    7.154963] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154963] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154966] EDAC MC: UMC8 chip selects:
[    7.154966] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154967] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154970] EDAC MC: UMC9 chip selects:
[    7.154970] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154970] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154973] EDAC MC: UMC10 chip selects:
[    7.154974] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154974] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.154977] EDAC MC: UMC11 chip selects:
[    7.154977] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.154977] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155298] EDAC MC1: Giving out device to module amd64_edac controller F1Ah: DEV 0000:00:19.3
(INTERRUPT)
```

```
[    7.155300] EDAC amd64: F1Ah detected (node 1).
[    7.155302] EDAC MC: UMC0 chip selects:
[    7.155302] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155303] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155305] EDAC MC: UMC1 chip selects:
[    7.155306] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155306] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155309] EDAC MC: UMC2 chip selects:
[    7.155309] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155309] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155312] EDAC MC: UMC3 chip selects:
[    7.155312] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155312] EDAC amd64: MC: 2: 32768MB 3:     0MB
[    7.155315] EDAC MC: UMC4 chip selects:
[    7.155315] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155316] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155318] EDAC MC: UMC5 chip selects:
[    7.155318] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155319] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155321] EDAC MC: UMC6 chip selects:
[    7.155322] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155322] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155324] EDAC MC: UMC7 chip selects:
[    7.155325] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155325] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155328] EDAC MC: UMC8 chip selects:
[    7.155328] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155328] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155331] EDAC MC: UMC9 chip selects:
[    7.155331] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155331] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155334] EDAC MC: UMC10 chip selects:
[    7.155334] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155334] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155337] EDAC MC: UMC11 chip selects:
[    7.155337] EDAC amd64: MC: 0:     0MB 1:     0MB
[    7.155338] EDAC amd64: MC: 2:     0MB 3:     0MB
[    7.155338] AMD64 EDAC driver v3.5.0
```

## 5.2 - Error Injection

The `mce-inject` tool provides valuable ECC diagnostics and debugging by simulating machine check errors.

```
# modprobe  mce-inject

[165070.534144] mce: Platform does not allow *hardware* error injection. Try using APEI EINJ instead.
[165070.534163] mce: Machine check injector initialized
```

If you want to generate real hardware errors, then you will have to use another interface, such as the EINJ interface or the AMD RAS error injection tool.

Here is an error injection example:

```
# cd /sys/kernel/debug/apei/einj
# cat available_error_type            # See which errors can be injected
0x00000002    Processor Uncorrectable non-fatal
0x00000008    Memory Correctable
0x00000010    Memory Uncorrectable non-fatal
# echo 0x12345000 > param1            # Set memory address for injection
# echo 0xfffffffffffff000 > param2            # Mask - anywhere in this page
# echo 0x8 > error_type                       # Choose correctable memory error
# echo 1 > error_inject                       # Inject now
You should see something like this in dmesg:
[22715.830801] EDAC sbridge MC3: HANDLING MCE MEMORY ERROR
```

```
[22715.834759] EDAC sbridge MC3: CPU 0: Machine Check Event: 0 Bank 7: 8c00004000010090
[22715.834759] EDAC sbridge MC3: TSC 0
[22715.834759] EDAC sbridge MC3: ADDR 12345000 EDAC sbridge MC3: MISC 144780c86
[22715.834759] EDAC sbridge MC3: PROCESSOR 0:306e7 TIME 1422553404 SOCKET 0 APIC 0
[22716.616173] EDAC MC3: 1 CE memory read error on CPU_SrcID#0_Channel#0_DIMM#0 (channel:0 slot:0
page:0x12345 offset:0x0 grain:32 syndrome:0x0 -  area:DRAM err_code:0001:0090 socket:0 channel_mask:1
rank:0)
```

Please see the following resources for additional information:

- APEI Error INJection*
- RAS Error Injection Tool Platform Guidance for Socket SP5 Family 1Ah Models 00h–0Fh and Models 10h–1Fh Processors (login required)
- AMD RAS Error Injection Test (login required)
  - Linux
  - Windows
- AMD RAS Error Injection Test for Family 19h Models 10h-1Fh Processors Training Guide (login required)

# Chapter 6: SPECpower and STREAM

Please see the *Optimizations and Tuning* appendix in AMD Power and Performance Optimization Guide for Family 19h Models 10h–1Fh.

## 6.1 - STREAM using Spack

STREAM tests the maximum memory bandwidth of a core or entire CPU. Please see the *High Performance Computing (HPC) Tuning Guide for AMD EPYC™ 9005 Series Processors* (available from the AMD Documentation Hub) for instructions on how to build the STREAM benchmark using Spack.

THIS PAGE INTENTIONALLY LEFT BLANK.

# Chapter 7: Resources

- Memory Population Guidelines for AMD Family 1Ah Models 00h–0Fh and Models 10h–1Fh Socket SP5 Processors - Login required; please review the latest version if multiple versions are present.
- Socket SP5/SP6 Platform NUMA Topology for AMD Family 1Ah Models 00h–0Fh and Models 10h–1Fh - Login required; please review the latest version if multiple versions are present.
- Add support for Large Increment per Cycle Events*
- Product Documentation for Red Hat Enterprise Linux 9*
- Red Hat Enterprise Linux 9 - Configuring and Managing Virtualization*
- Red Hat Enterprise Linux 9.0 Release Notes*
- Red Hat Enterprise Linux 7: Optimizing Memory System Performance*
- Enabling AMD Secure Encrypted Virtualization in RHEL 8*
- AMD Power and Performance Optimization Guide for Family 19h Models 10h–1Fh (login required)
- From the AMD Documentation Hub:
  - *High Performance Computing (HPC) Tuning Guide for AMD EPYC™ 9005 Series Processors*
  - *Overview of AMD EPYC™ 9005 Series Processors Architecture*
  - *BIOS & Workload Tuning Guide for AMD EPYC™ 9005 Series Processors*
  - *Kubernetes® Containers Tuning Guide for AMD EPYC™ 9005 Series Processors*
  - *OpenShift® Containers Tuning Guide for AMD EPYC™ 9005 Series Processors*
- Configuring PCI passthrough*
- OpenShift Container Platform 4.15 Documentation*

THIS PAGE INTENTIONALLY LEFT BLANK.

# Chapter 8: AMD-Specific Kernel Features & Fixes

## 8.1 - IOMMU v2 page table 5 level paging:

`iommu/amd` adds five-level guest page table support.

## 8.2 - Generic IO Page Table Framework Support for v2 Page Table

A new usage model was created for the v2 page table where it can be used to implement support for DMA-API by adopting the generic IO page table framework.

Kernel command line: `amd_iommu=pgtbl_v2`

The following new features were added:

- Initial support for AMD IOMMU v2 page table.
- Add command-line option to enable different page tables.
- Add support for using AMD IOMMU v2 page table for DMA-API.
- Add support for Guest IO protection.
- Update sanity check when enable PRI/ATS for IOMMU v1 table.
- Refactor `amd_iommu_domain_enable_v2` to remove locking.
- Add `map/unmap_pages() iommu_domain_ops` callback support.
- Implement `unmap_pages io_pgtable_ops` callback in `iommu/amd/io-pgtable`.
- Implement `map_pages io_pgtable_ops` callback in `iommu/amd/io-pgtable`.

AMD IOMMU supports various page size mapping:

- V1 page table supports 4K, 8K... 4G
- V1 supports page size up to 512G.
- V2 page table support 4K, 2M, and 1G
- Linux Guest Kernel supports the v2 page table only.

## 8.3 - AVIC Interrupt Remapping Improvements

The following improvements were made to the AMD IOMMU:

- Improved interrupt remapping table invalidation.
- Switch `amd_iommu_update_ga()` to use `modify_irte_ga()`.
- Invalidate IRT when IRTE caching is disabled
- Introduce Disable IRTE Caching support.
- Remove the unused `struct amd_ir_data.ref`.

## 8.4 - AMD Cache Computation Fix

All AMD architectures cache details will be computed based on `_cpuid_ `0x8000_001D` and the reference to `_cpuid_ `0x8000_0006` will be zeroed out for future architectures.

## 8.5 - Predictive Store Forwarding Disable (PSFD)

STLF (Store-To-Load Forwarding) occurs after the address of both the load and store are calculated and determined to match.

PSF (Predictive Store Forwarding) expands on this by speculating on the relationship between loads and stores without waiting for the address calculation to complete. With PSF, the CPU learns over time the relationship between loads and stores. If STLF typically occurs between a particular store and load, then the CPU will remember this. In typical code, PSF provides a performance benefit by speculating on the load result and allowing later instructions to begin execution sooner than they otherwise would be able to. See Security Analysis of AMD Predictive Store Forwarding for additional information.

PSF uses two hardware control bits:

- **MSR 48h bit 2:** Speculative Store Bypass (SSBD); disables both PSF and SSBD.
- **MSR 48h bit 7:** Predictive Store Forwarding Disable (PSFD); disables PSF only. PSFD may be desirable for software that leverages the speculative behavior of PSF but desires a smaller performance impact than setting SSBD. Support for PSFD is indicated in `CPUID Fn8000_0008 EBX[28]`. All processors that support PSF will also support PSFD.

Setting either bit disables PSF. These bits are controllable on a per-thread basis in an SMT system. By default, both SSBD and PSFD are 0, meaning that the speculation features are enabled.

The current Linux kernel does not have the interface to enable/disable PSFD. The plan is to expose the PSFD technology to KVM so that the guest kernel can make use of it if desired.

## 8.6 - Other Features & Fixes

- The current AVIC implementation cannot support encrypted memory. Be sure to inhibit AVIC for SEV-enabled guest.
- `iommu/amd`: This code has been restructured to free page tables.
- The OS cannot boot when enabling SME in a UEFI setup and appending `mem_encrypt=on`.
- Potential host crash introduced by SEV-SNP guest support - Fixed.

## 8.7 - PCIe Gens Support

```
#lspci -vvv (Speed 32GT/s)
00:07.1 PCI bridge: Adced Micro Devices, Inc. [AMD] Device 14a7 (rev 01) (prog-if 00 [Normal decode])
        Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B-
DisINTx+
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR-
INTx-
        Latency: 0, Cache Line Size: 64 bytes
        Interrupt: pin A routed to IRQ 47
        NUMA node: 0
        IOMMU group: 25
        Bus: primary=00, secondary=02, subordinate=02, sec-latency=0
        I/O behind bridge: 0000f000-00000fff [disabled]
        Memory behind bridge: fa000000-fa2fffff [size=3M]
        Prefetchable memory behind bridge: 000002807ff00000-000002807fffffff [size=1M]
        Secondary status: 66MHz- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- <SERR- <PERR-
        BridgeCtl: Parity- SERR+ NoISA- VGA- VGA16+ MAbort- >Reset- FastB2B-
                PriDiscTmr- SecDiscTmr- DiscTmrStat- DiscTmrSERREn-
        Capabilities: [50] Power Management version 3
                Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0+,D1-,D2-,D3hot+,D3cold+)
                Status: D0 NoSoftRst- PME-Enable- DSel=0 DScale=0 PME-
        Capabilities: [58] Express (v2) Root Port (Slot-), MSI 00
                DevCap: MaxPayload 512 bytes, PhantFunc 0
                        ExtTag+ RBE+
                DevCtl: CorrErr- NonFatalErr- FatalErr- UnsupReq-
                        RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+
                        MaxPayload 128 bytes, MaxReadReq 512 bytes
                DevSta: CorrErr- NonFatalErr- FatalErr- UnsupReq- AuxPwr- TransPend-
                LnkCap: Port #0, Speed 32GT/s, Width x16, ASPM L0s L1, Exit Latency L0s <64ns, L1 <1us
                        ClockPM- Surprise- LLActRep+ BwNot+ ASPMOptComp+
                LnkCtl: ASPM Disabled; RCB 64 bytes, Disabled- CommClk+
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
                LnkSta: Speed 32GT/s (ok), Width x16 (ok)
                        TrErr- Train- SlotClk+ DLActive+ BWMgmt+ ABWMgmt-
                RootCap: CRSVisible+
                RootCtl: ErrCorrectable- ErrNon-Fatal- ErrFatal- PMEIntEna+ CRSVisible+
                RootSta: PME ReqID 0000, PMEStatus- PMEPending-
                DevCap2: Completion Timeout: Not Supported, TimeoutDis- NROPrPrP- LTR-
                         10BitTagComp+ 10BitTagReq- OBFF Not Supported, ExtFmt- EETLPPrefix-
                         EmergencyPowerReduction Not Supported, EmergencyPowerReductionInit-
                         FRS- LN System CLS Not Supported, TPHComp+ ExtTPHComp- ARIFwd-
                         AtomicOpsCap: Routing- 32bit- 64bit- 128bitCAS-
                DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis- LTR- OBFF Disabled, ARIFwd-
                         AtomicOpsCtl: ReqEn- EgressBlck-
                LnkCap2: Supported Link Speeds: 2.5-32GT/s, Crosslink- Retimer+ 2Retimers+ DRS-
                LnkCtl2: Target Link Speed: 32GT/s, EnterCompliance- SpeedDis-
                         Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
                         Compliance De-emphasis: -6dB
.............
.............
.....................
```

## 8.8 - PCIe Multiple Segments Support (RHEL 9.2 and Later)

AMD 9xx5 and 9xx4 systems can support multiple PCI segments, where each segment contains one or more IOMMU instances. However, an IOMMU instance can only support a single PCI segment. Legacy code assumes a system contains only one PCI segment (segment 0) and creates global data structures, such as device table, lookup table, etc. This introduces per-PCI-segment data structure, which contains device table, alias table, etc.

For each PCI segment, all IOMMUs share the same data structure. Global data structures like device table, alias table, etc. are removed. However, in systems with a single PCI segment (e.g., PCI segment ID is zero), the IOMMU driver allocates one PCI segment data structure, which will be shared by all IOMMUs. For example:

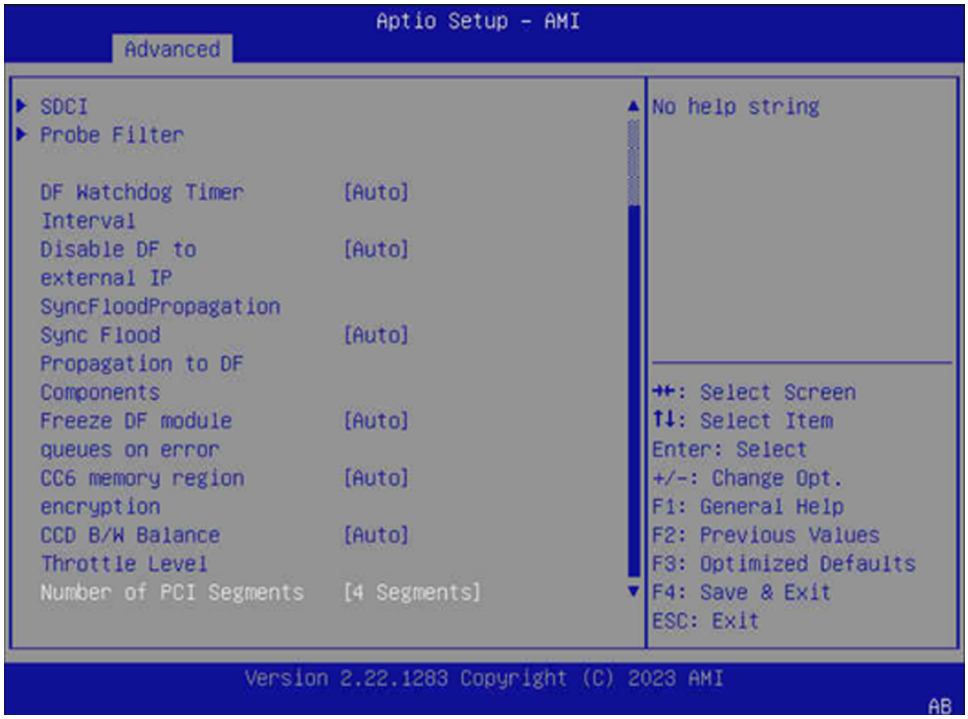


*Figure 8-1: PCIe multiple segments support*

```
4 Domains:

# dmesg | grep -i domain

[    0.625678] PCI: MMCONFIG for domain 0000 [bus 00-ff] at [mem 0x3ffb00000000-0x3ffb0fffffff] (base
0x3ffb00000000)
[    0.625681] PCI: MMCONFIG for domain 0001 [bus 00-ff] at [mem 0x3ffb10000000-0x3ffb1fffffff] (base
0x3ffb10000000)
[    0.625683] PCI: MMCONFIG for domain 0002 [bus 00-ff] at [mem 0x3ffb20000000-0x3ffb2fffffff] (base
0x3ffb20000000)
[    0.625684] PCI: MMCONFIG for domain 0003 [bus 00-ff] at [mem 0x3ffb30000000-0x3ffb3fffffff] (base
0x3ffb30000000)
[    0.649457] ACPI: PCI Root Bridge [S0D0] (domain 0003 [bus 00-ff])
[    0.656686] ACPI: PCI Root Bridge [S0D2] (domain 0000 [bus 00-ff])
[    0.664790] ACPI: PCI Root Bridge [PCI0] (domain 0001 [bus 00-ff])


# ls -al /sys/bus/pci/devices
total 0
drwxr-xr-x. 2 root root 0 Jul 13 15:22 .
drwxr-xr-x. 5 root root 0 Jul 13 15:22 ..
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:00.0 -> ../../../devices/pci0000:00/0000:00:00.0
```

```
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:00.2 -> ../../../devices/pci0000:00/0000:00:00.2
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:00.3 -> ../../../devices/pci0000:00/0000:00:00.3
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:01.0 -> ../../../devices/pci0000:00/0000:00:01.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:01.1 -> ../../../devices/pci0000:00/0000:00:01.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:02.0 -> ../../../devices/pci0000:00/0000:00:02.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:03.0 -> ../../../devices/pci0000:00/0000:00:03.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:03.1 -> ../../../devices/pci0000:00/0000:00:03.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:03.2 -> ../../../devices/pci0000:00/0000:00:03.2
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:03.3 -> ../../../devices/pci0000:00/0000:00:03.3
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:03.4 -> ../../../devices/pci0000:00/0000:00:03.4
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:04.0 -> ../../../devices/pci0000:00/0000:00:04.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:05.0 -> ../../../devices/pci0000:00/0000:00:05.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:05.1 -> ../../../devices/pci0000:00/0000:00:05.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:07.0 -> ../../../devices/pci0000:00/0000:00:07.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:07.1 -> ../../../devices/pci0000:00/0000:00:07.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:07.2 -> ../../../devices/pci0000:00/0000:00:07.2
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:14.0 -> ../../../devices/pci0000:00/0000:00:14.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:14.3 -> ../../../devices/pci0000:00/0000:00:14.3
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.0 -> ../../../devices/pci0000:00/0000:00:18.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.1 -> ../../../devices/pci0000:00/0000:00:18.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.2 -> ../../../devices/pci0000:00/0000:00:18.2
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.3 -> ../../../devices/pci0000:00/0000:00:18.3
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.4 -> ../../../devices/pci0000:00/0000:00:18.4
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.5 -> ../../../devices/pci0000:00/0000:00:18.5
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.6 -> ../../../devices/pci0000:00/0000:00:18.6
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:00:18.7 -> ../../../devices/pci0000:00/0000:00:18.7
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:03:00.0 -> ../../../devices/pci0000:00/0000:00:03.2/
0000:03:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:04:00.0 -> ../../../devices/pci0000:00/0000:00:03.3/
0000:04:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:06:00.0 -> ../../../devices/pci0000:00/0000:00:05.1/
0000:06:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:07:00.0 -> ../../../devices/pci0000:00/0000:00:07.1/
0000:07:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:07:00.1 -> ../../../devices/pci0000:00/0000:00:07.1/
0000:07:00.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:07:00.4 -> ../../../devices/pci0000:00/0000:00:07.1/
0000:07:00.4
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:07:00.5 -> ../../../devices/pci0000:00/0000:00:07.1/
0000:07:00.5
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:08:00.0 -> ../../../devices/pci0000:00/0000:00:07.2/
0000:08:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0000:08:00.1 -> ../../../devices/pci0000:00/0000:00:07.2/
0000:08:00.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:00.0 -> ../../../devices/pci0001:00/0001:00:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:00.2 -> ../../../devices/pci0001:00/0001:00:00.2
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:00.3 -> ../../../devices/pci0001:00/0001:00:00.3
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:01.0 -> ../../../devices/pci0001:00/0001:00:01.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:02.0 -> ../../../devices/pci0001:00/0001:00:02.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:03.0 -> ../../../devices/pci0001:00/0001:00:03.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:04.0 -> ../../../devices/pci0001:00/0001:00:04.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:05.0 -> ../../../devices/pci0001:00/0001:00:05.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:07.0 -> ../../../devices/pci0001:00/0001:00:07.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:00:07.1 -> ../../../devices/pci0001:00/0001:00:07.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:01:00.0 -> ../../../devices/pci0001:00/0001:00:07.1/
0001:01:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0001:01:00.1 -> ../../../devices/pci0001:00/0001:00:07.1/
0001:01:00.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:00.0 -> ../../../devices/pci0003:00/0003:00:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:00.2 -> ../../../devices/pci0003:00/0003:00:00.2
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:00.3 -> ../../../devices/pci0003:00/0003:00:00.3
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:01.0 -> ../../../devices/pci0003:00/0003:00:01.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:02.0 -> ../../../devices/pci0003:00/0003:00:02.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:03.0 -> ../../../devices/pci0003:00/0003:00:03.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:03.1 -> ../../../devices/pci0003:00/0003:00:03.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:03.2 -> ../../../devices/pci0003:00/0003:00:03.2
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:03.3 -> ../../../devices/pci0003:00/0003:00:03.3
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:03.4 -> ../../../devices/pci0003:00/0003:00:03.4
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:04.0 -> ../../../devices/pci0003:00/0003:00:04.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:05.0 -> ../../../devices/pci0003:00/0003:00:05.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:05.1 -> ../../../devices/pci0003:00/0003:00:05.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:05.2 -> ../../../devices/pci0003:00/0003:00:05.2
```

```
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:07.0 -> ../../../devices/pci0003:00/0003:00:07.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:00:07.1 -> ../../../devices/pci0003:00/0003:00:07.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:05:00.0 -> ../../../devices/pci0003:00/0003:00:05.1/
0003:05:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:06:00.0 -> ../../../devices/pci0003:00/0003:00:05.1/
0003:05:00.0/0003:06:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:07:00.0 -> ../../../devices/pci0003:00/0003:00:05.2/
0003:07:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:07:00.1 -> ../../../devices/pci0003:00/0003:00:05.2/
0003:07:00.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:08:00.0 -> ../../../devices/pci0003:00/0003:00:07.1/
0003:08:00.0
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:08:00.1 -> ../../../devices/pci0003:00/0003:00:07.1/
0003:08:00.1
lrwxrwxrwx. 1 root root 0 Jul 13 15:22 0003:08:00.4 -> ../../../devices/pci0003:00/0003:00:07.1/
0003:08:00.4
```

## 8.9 - CXL Memory

Check the ACPI STRAT tables for the memory range.

- The CXL memory node must show up as a separate node. (Node 2 appears in the example below.)
- The CXL memory node does not have CPUs connected to it.
- The address range found must be reasonable and must match the size of the memory provided by the CXL memory device.

```
# lspci | grep -i cxl
1f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)
3f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)
5f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)
7f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)

CXL Legacy Support
NUMA topology
ACPI SRAT
Check ACPI SRAT table entries for the mem range:
•The CXL memory node must show up as a separate node.
•The CXL memory node does not have CPUs connected to it.
•The address range found is reasonable and matches the size of the mem provided by the CXL mem device.


# dmesg | grep -i srat

[    0.007227] ACPI: SRAT 0x00000000A4261000 0024F8 (v03 AMD    AmdTable 00000001 AMD  00000001)
[    0.007253] ACPI: Reserving SRAT table memory at [mem 0xa4261000-0xa42634f7]
[    0.007337] SRAT: PXM 0 -> APIC 0x0000 -> Node 0
[    0.007338] SRAT: PXM 0 -> APIC 0x0001 -> Node 0
[    0.007339] SRAT: PXM 0 -> APIC 0x0002 -> Node 0
[    0.007339] SRAT: PXM 0 -> APIC 0x0003 -> Node 0
…..
……
……
[    0.007354] SRAT: PXM 0 -> APIC 0x002c -> Node 0
[    0.007354] SRAT: PXM 0 -> APIC 0x002d -> Node 0
[    0.007354] SRAT: PXM 0 -> APIC 0x002e -> Node 0
[    0.007355] SRAT: PXM 0 -> APIC 0x002f -> Node 0
[    0.007355] SRAT: PXM 0 -> APIC 0x0030 -> Node 0
[    0.007355] SRAT: PXM 0 -> APIC 0x0031 -> Node 0
[    0.007356] SRAT: PXM 0 -> APIC 0x0032 -> Node 0
[    0.007356] SRAT: PXM 0 -> APIC 0x0033 -> Node 0
[    0.007357] SRAT: PXM 0 -> APIC 0x0034 -> Node 0
[    0.007357] SRAT: PXM 0 -> APIC 0x0035 -> Node 0
[    0.007358] SRAT: PXM 0 -> APIC 0x0036 -> Node 0
[    0.007359] SRAT: PXM 0 -> APIC 0x0037 -> Node 0
[    0.007359] SRAT: PXM 0 -> APIC 0x0038 -> Node 0
```

```
[    0.007359] SRAT: PXM 0 -> APIC 0x0039 -> Node 0
[    0.007360] SRAT: PXM 0 -> APIC 0x003a -> Node 0
[    0.007360] SRAT: PXM 0 -> APIC 0x003b -> Node 0
[    0.007360] SRAT: PXM 0 -> APIC 0x003c -> Node 0
[    0.007361] SRAT: PXM 0 -> APIC 0x003d -> Node 0
[    0.007361] SRAT: PXM 0 -> APIC 0x003e -> Node 0
[    0.007362] SRAT: PXM 0 -> APIC 0x003f -> Node 0
[    0.007362] SRAT: PXM 0 -> APIC 0x0040 -> Node 0
[    0.007362] SRAT: PXM 0 -> APIC 0x0041 -> Node 0
[    0.007363] SRAT: PXM 0 -> APIC 0x0042 -> Node 0
[    0.007363] SRAT: PXM 0 -> APIC 0x0043 -> Node 0
[    0.007363] SRAT: PXM 0 -> APIC 0x0044 -> Node 0
[    0.007364] SRAT: PXM 0 -> APIC 0x0045 -> Node 0
[    0.007364] SRAT: PXM 0 -> APIC 0x0046 -> Node 0
[    0.007364] SRAT: PXM 0 -> APIC 0x0047 -> Node 0
[    0.007365] SRAT: PXM 0 -> APIC 0x0048 -> Node 0
[    0.007365] SRAT: PXM 0 -> APIC 0x0049 -> Node 0
[    0.007365] SRAT: PXM 0 -> APIC 0x004a -> Node 0
[    0.007366] SRAT: PXM 0 -> APIC 0x004b -> Node 0
….
……
……
[    0.007468] SRAT: PXM 1 -> APIC 0x01ad -> Node 1
[    0.007469] SRAT: PXM 1 -> APIC 0x01ae -> Node 1
[    0.007469] SRAT: PXM 1 -> APIC 0x01af -> Node 1
[    0.007469] SRAT: PXM 1 -> APIC 0x01b0 -> Node 1
[    0.007470] SRAT: PXM 1 -> APIC 0x01b1 -> Node 1
[    0.007470] SRAT: PXM 1 -> APIC 0x01b2 -> Node 1
[    0.007470] SRAT: PXM 1 -> APIC 0x01b3 -> Node 1
[    0.007471] SRAT: PXM 1 -> APIC 0x01b4 -> Node 1
[    0.007471] SRAT: PXM 1 -> APIC 0x01b5 -> Node 1
[    0.007472] SRAT: PXM 1 -> APIC 0x01b6 -> Node 1
[    0.007472] SRAT: PXM 1 -> APIC 0x01b7 -> Node 1
[    0.007472] SRAT: PXM 1 -> APIC 0x01b8 -> Node 1
[    0.007473] SRAT: PXM 1 -> APIC 0x01b9 -> Node 1
[    0.007473] SRAT: PXM 1 -> APIC 0x01ba -> Node 1
[    0.007473] SRAT: PXM 1 -> APIC 0x01bb -> Node 1
[    0.007474] SRAT: PXM 1 -> APIC 0x01bc -> Node 1
[    0.007474] SRAT: PXM 1 -> APIC 0x01bd -> Node 1
[    0.007474] SRAT: PXM 1 -> APIC 0x01be -> Node 1
[    0.007475] SRAT: PXM 1 -> APIC 0x01bf -> Node 1
[    0.007479] ACPI: SRAT: Node 0 PXM 0 [mem 0x00000000-0x0009ffff]
[    0.007482] ACPI: SRAT: Node 0 PXM 0 [mem 0x000c0000-0xafffffff]
[    0.007483] ACPI: SRAT: Node 0 PXM 0 [mem 0x100000000-0x84fffffff]
[    0.007484] ACPI: SRAT: Node 1 PXM 1 [mem 0x850000000-0x104fffffff]
[    0.007484] ACPI: SRAT: Node 2 PXM 2 [mem 0x1050000000-0x904fffffff]
```

### 8.9.1 - ACPI SRAT/SLIT (numactl)

Check the ACPI SLIT entries (node distances) and the node's CPU list using `numactl`:

- The CXL memory node does not have CPUs connected to it (CPU list is empty).
- The CPU-CXL node distance is greater than CPU-to-CPU distances (typically 50).

```
# dmesg | grep -i numa
[    0.007489] NUMA: Initialized distance table, cnt=3
[    0.007491] NUMA: Node 0 [mem 0x00000000-0x0009ffff] + [mem 0x000c0000-0xafffffff] -> [mem 0x00000000-
0xafffffff]
[    0.007493] NUMA: Node 0 [mem 0x00000000-0xafffffff] + [mem 0x100000000-0x84fffffff] -> [mem 0x00000000-
0x84fffffff]
[    0.225981] mempolicy: Enabling automatic NUMA balancing. Configure with numa_balancing= or the
kernel.numa_balancing sysctl
[    4.518775] pci_bus 0000:60: on NUMA node 0
[    4.520095] pci_bus 0000:40: on NUMA node 0
[    4.523849] pci_bus 0000:00: on NUMA node 0
[    4.527057] pci_bus 0000:20: on NUMA node 0
[    4.528968] pci_bus 0000:e0: on NUMA node 1
```

```
[    4.530528] pci_bus 0000:c0: on NUMA node 1
[    4.532661] pci_bus 0000:80: on NUMA node 1
[    4.534499] pci_bus 0000:a0: on NUMA node 1
[    4.535136] pci_bus 0000:7f: Unknown NUMA node; performance will be reduced
[    4.535725] pci_bus 0000:5f: Unknown NUMA node; performance will be reduced
[    4.536286] pci_bus 0000:1f: Unknown NUMA node; performance will be reduced
[    4.536859] pci_bus 0000:3f: Unknown NUMA node; performance will be reduced

# numactl -H

available: 3 nodes (0-2)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 192 193 194 195 196 197 198 199
200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226
227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253
254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280
281 282 283 284 285 286 287
node 0 size: 31425 MB
node 0 free: 28230 MB
node 1 cpus: 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146
147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173
174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 288 289 290 291 292 293 294 295 296
297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323
324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350
351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377
378 379 380 381 382 383
node 1 size: 32158 MB
node 1 free: 30406 MB
node 2 cpus:
node 2 size: 516017 MB
node 2 free: 514964 MB
node distances:
node   0   1   2
  0:  10  32  50
  1:  32  10  60
  2:  255  255  10


System Address Map (e820 table/GetMemoryMap())
Check e820 memory map in the early boot log. The CXL memory range needs to show up, e.g. as
0x0000001050000000-0x000000904fffffff
Method 1 (dmesg)
Using early boot log:


# dmesg | grep -iw e820.*usable

[    0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009ffff] usable
[    0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000a0bdbfff] usable
[    0.000000] BIOS-e820: [mem 0x00000000a57ff000-0x00000000a7ffcfff] usable
[    0.000000] BIOS-e820: [mem 0x0000001000000000-0x000000104d0bffff] usable
[    0.000000] BIOS-e820: [mem 0x0000001050000000-0x000000904fffffff] usable
[    0.000000] e820: update [mem 0x991da018-0x991ea057] usable ==> usable
[    0.000000] e820: update [mem 0x991da018-0x991ea057] usable ==> usable
[    0.000000] e820: update [mem 0x991bc018-0x991d9857] usable ==> usable
[    0.000000] e820: update [mem 0x991bc018-0x991d9857] usable ==> usable
[    0.000271] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[    0.000279] e820: remove [mem 0x000a0000-0x000fffff] usable
[    0.003422] e820: update [mem 0xb0000000-0xffffffff] usable ==> reserved
[    0.006596] e820: update [mem 0x991fc000-0x991fcfff] usable ==> reserved
[    0.012657] e820: update [mem 0x9af1c000-0x9af6efff] usable ==> reserved
```

## 8.9.2 - ACPI SRAT/SLIT (/proc/iomap)

To check the ACPI SLIT entries (node distances) and the node's CPU list using `/proc` filesystem:

```
# cat /proc/iomem | grep RAM
00001000-0009ffff : System RAM
00100000-991bc017 : System RAM
991bc018-991d9857 : System RAM
991d9858-991da017 : System RAM
991da018-991ea057 : System RAM
991ea058-991fbfff : System RAM
991fd000-9af1bfff : System RAM
9af6f000-a0bdbfff : System RAM
a57ff000-a7ffcfff : System RAM
100000000-104d0bfffff : System RAM
1050000000-904fffffff : System RAM
```

Determine the address range of the CXL memory (see "ACPI SRAT/SLIT (numactl)" on page 49). It must show up in the BIOS-e820 entries listed in `dmesg` or as `System RAM` in `/proc/iomem`.

ACPI SRAT only provides limited knowledge to determine the CXL memory range of a device. You must also use side channels or system knowledge. Future BIOS releases my show the address range as `soft reserved` instead of `usable`. Adjust test patterns accordingly.

Only the early boot messages show the memory map provided by the BIOS. The kernel may decide to change the mappings, such as when using the `mem= kernel` boot parameter whee the memory type of certain address ranges change to `reserved`. That is, `/proc` entries or later boot messages may not match the BIOS mappings.

```
# lspci | grep -i cxl

1f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)
3f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)
5f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)
7f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02)

# lspci -vvv -s 5f:00.0

5f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02) (prog-if 10 [CXL
Memory Device (CXL 2.x)])
        Subsystem: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963
        Control: I/O- Mem+ BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B-
DisINTx-
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR-
INTx-
        Interrupt: pin A routed to IRQ 194
        IOMMU group: 66
        Region 0: Memory at d1f00000 (32-bit, non-prefetchable) [size=128K]
        Region 2: Memory at 20020f00000 (64-bit, prefetchable) [size=8K]
        Capabilities: [80] Express (v2) Root Complex Integrated Endpoint, MSI 00
                DevCap: MaxPayload 512 bytes, PhantFunc 0
                        ExtTag+ RBE+ FLReset-
                DevCtl: CorrErr- NonFatalErr- FatalErr- UnsupReq-
                        RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+
                        MaxPayload 512 bytes, MaxReadReq 512 bytes
                DevSta: CorrErr- NonFatalErr- FatalErr- UnsupReq- AuxPwr- TransPend-
                DevCap2: Completion Timeout: Not Supported, TimeoutDis+ NROPrPrP- LTR-
                         10BitTagComp+ 10BitTagReq- OBFF Not Supported, ExtFmt+ EETLPPrefix-
                         EmergencyPowerReduction Not Supported, EmergencyPowerReductionInit-
                         FRS-
                         AtomicOpsCap: 32bit- 64bit- 128bitCAS-
                DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis- LTR- OBFF Disabled,
                         AtomicOpsCtl: ReqEn-
        Capabilities: [e0] MSI: Enable- Count=1/1 Maskable- 64bit+
                Address: 0000000000000000  Data: 0000
        Capabilities: [f8] Power Management version 3
                Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
                Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-
        Capabilities: [100 v1] Vendor Specific Information: ID=1556 Rev=1 Len=008 <?>
```

```
…….
……..
……..
                ARICtl: MFVC- ACS-, Function Group: 0
        Capabilities: [1e0 v1] Data Link Feature <?>
        Capabilities: [200 v2] Advanced Error Reporting
                UESta:  DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq-
ACSViol-
                UEMsk:  DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq-
ACSViol-
                UESvrt: DLP+ SDES- TLP- FCP+ CmpltTO- CmpltAbrt- UnxCmplt- RxOF+ MalfTLP+ ECRC- UnsupReq-
ACSViol-
                CESta:  RxErr- BadTLP- BadDLLP- Rollover- Timeout- AdvNonFatalErr-
                CEMsk:  RxErr- BadTLP- BadDLLP- Rollover- Timeout- AdvNonFatalErr+
                AERCap: First Error Pointer: 00, ECRCGenCap- ECRCGenEn- ECRCChkCap- ECRCChkEn-
                        MultHdrRecCap- MultHdrRecEn- TLPPfxPres- HdrLogCap-
                HeaderLog: 00000000 00000000 00000000 00000000
        Capabilities: [450 v1] Extended Capability ID 0x2e
        Capabilities: [480 v1] Virtual Channel
                Caps:   LPEVC=0 RefClk=100ns PATEntryBits=1
                Arb:    Fixed- WRR32- WRR64- WRR128-
                Ctrl:   ArbSelect=Fixed
                Status: InProgress-
                VC0:    Caps:   PATOffset=00 MaxTimeSlots=1 RejSnoopTrans-
                        Arb:    Fixed- WRR32- WRR64- WRR128- TWRR128- WRR256-
                        Ctrl:   Enable+ ID=0 ArbSelect=Fixed TC/VC=01
                        Status: NegoPending- InProgress-
        Capabilities: [500 v1] Designated Vendor-Specific: Vendor=1e98 ID=0000 Rev=1 Len=56: CXL
                CXLCap: Cache- IO+ Mem+ Mem HW Init+ HDMCount 1 Viral+
                CXLCtl: Cache- IO+ Mem+ Cache SF Cov 0 Cache SF Gran 0 Cache Clean- Viral-
                CXLSta: Viral-
…….
……..
……..
        Capabilities: [5d0 v1] Designated Vendor-Specific: Vendor=1e98 ID=000a Rev=0 Len=28 <?>
        Kernel driver in use: cxl_pci
        Kernel modules: cxl_pci
…….
……..
……..

# lspci -vvv -s 1f:00.0

1f:00.0 CXL: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963 (rev 02) (prog-if 10 [CXL
Memory Device (CXL 2.x)])
        Subsystem: Samsung Electronics Co Ltd NVMe SSD Controller SM961/PM961/SM963
        Control: I/O- Mem+ BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B-
DisINTx-
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR-
INTx-
        Interrupt: pin A routed to IRQ 191
        IOMMU group: 67
        Region 0: Memory at fdf00000 (32-bit, non-prefetchable) [size=128K]
        Region 2: Memory at 28080f00000 (64-bit, prefetchable) [size=8K]
        Capabilities: [80] Express (v2) Root Complex Integrated Endpoint, MSI 00
                DevCap: MaxPayload 512 bytes, PhantFunc 0
                        ExtTag+ RBE+ FLReset-
                DevCtl: CorrErr- NonFatalErr- FatalErr- UnsupReq-
                        RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+
                        MaxPayload 512 bytes, MaxReadReq 512 bytes
                DevSta: CorrErr- NonFatalErr- FatalErr- UnsupReq- AuxPwr- TransPend-
                DevCap2: Completion Timeout: Not Supported, TimeoutDis+ NROPrPrP- LTR-
                         10BitTagComp+ 10BitTagReq- OBFF Not Supported, ExtFmt+ EETLPPrefix-
                         EmergencyPowerReduction Not Supported, EmergencyPowerReductionInit-
                         FRS-
                         AtomicOpsCap: 32bit- 64bit- 128bitCAS-
…….
……..
……..
        Capabilities: [500 v1] Designated Vendor-Specific: Vendor=1e98 ID=0000 Rev=1 Len=56: CXL
                CXLCap: Cache- IO+ Mem+ Mem HW Init+ HDMCount 1 Viral+
                CXLCtl: Cache- IO+ Mem+ Cache SF Cov 0 Cache SF Gran 0 Cache Clean- Viral-
```

```
                CXLSta: Viral-
…….
……..
……..
        Capabilities: [5d0 v1] Designated Vendor-Specific: Vendor=1e98 ID=000a Rev=0 Len=28 <?>
        Kernel driver in use: cxl_pci
        Kernel modules: cxl_pci
```

### 8.9.3 - ndctl/daxctl

The `ndctl` utility is used to manage persistent memory devices within the system. The `daxctl` utility manages device-dax instances.

```
# yum install ndctl

Updating Subscription Management repositories.
Last metadata expiration check: 0:38:39 ago on Tue 14 Mar 2023 03:46:09 PM EDT.
Dependencies resolved.
================================================================================================================
================================================================================================================
============================
 Package                            Architecture                      Version
Repository                                                              Size
================================================================================================================
================================================================================================================
============================
Installing:
 ndctl                                          x86_64                                                  71.1-
7.el9                                  rhel-9-for-x86_64-baseos-rpms
193 k
Installing dependencies:
 daxctl-libs                                    x86_64                                                  71.1-
7.el9                                 rhel-9-for-x86_64-baseos-rpms
43 k
 ndctl-libs                                     x86_64                                                  71.1-
7.el9                                 rhel-9-for-x86_64-baseos-rpms
85 k

Transaction Summary
================================================================================================================
================================================================================================================
============================
Install  3 Packages

Total download size: 321 k
Installed size: 630 k
Is this ok [y/N]: y
Downloading Packages:
(1/3): daxctl-libs-71.1-7.el9.x86_64.rpm
107 kB/s | 43 kB     00:00
(2/3): ndctl-libs-71.1-7.el9.x86_64.rpm
203 kB/s |  85 kB     00:00
(3/3): ndctl-71.1-7.el9.x86_64.rpm



# yum install daxctl

Updating Subscription Management repositories.
Last metadata expiration check: 0:00:12 ago on Tue 14 Mar 2023 08:02:50 PM EDT.
Dependencies resolved.
================================================================================================================
================================================================================================================
============================
 Package                          Architecture                      Version
Repository                                                              Size
================================================================================================================
================================================================================================================
============================
```

```
Installing:
 daxctl                                          x86_64                                                                  71.1-
7.el9                                  rhel-9-for-x86_64-appstream-rpms
65 k

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

Total download size: 65 k
Installed size: 109 k
Is this ok [y/N]: y
Downloading Packages:
daxctl-71.1-7.el9.x86_64.rpm
171 kB/s |  65 kB     00:00
--------------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------------------------------------
----------------------------
Total
170 kB/s |  65 kB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
 Preparing    :
1/1
 Installing   : daxctl-71.1-7.el9.x86_64
1/1
 Running scriptlet: daxctl-71.1-7.el9.x86_64
1/1
 Verifying    : daxctl-71.1-7.el9.x86_64
1/1
Installed products updated.

Installed:
  daxctl-71.1-7.el9.x86_64

Complete!
```

## 8.10 - FRU Text in MCA" Feature (RHEL 9.2 and Later)

A new "FRU Text in MCA" feature is defined where the Field Replaceable Unit (FRU) Text for a device is represented by a string in the new `MCA_SYND1` and `MCA_SYND2` registers. This feature is supported per MCA bank, and it is advertised by the `McaFruTextInMca bit (MCA_CONFIG[9])`.

The FRU Text is populated dynamically for each individual error state (`MCA_STATUS`, `MCA_ADDR`, et al.). This handles the case where an MCA bank covers multiple devices, for example, a Unified Memory Controller (UMC) bank that manages two DIMMs.

Print the FRU Text string, if available, when decoding an MCA error.

## 8.11 - AVIC And X2AVIC Enablement (RHEL 9.2 and Later)

The AMD Virtual Interrupt Controller or AVIC hardware virtualizes the local APIC registers of each vCPU via the virtual APIC (vAPIC) backing page. X2AVIC is an extension to the existing AVIC feature. X2AVIC virtualizes X2APIC accesses, which increases the addressability for logical and physical destination modes to support an increased number of CPUs. X2AVIC mode accesses these via the MSR interface, unlike memory access to APIC registers in AVIC mode. Verify that these features are enabled in BIOS before proceeding.

A new feature bit was introduced for virtualized x2APIC (x2AVIC) in `CPUID_Fn8000000A_EDX` [SVM Revision and Feature Identification].

Add CPUID check for the x2APIC virtualization (x2AVIC) feature. If available, the SVM driver can support both AVIC and x2AVIC modes, when loading the `kvm_amd` driver with `avic=1.` The operating mode will be determined at runtime depending on the guest APIC mode.

The `param avic` module enables both xAPIC and x2APIC modes.

Hypervisor can support both xAVIC and x2AVIC in the same guest, and the mode can be switched at runtime.

On the host:

```
modprobe -r kvm_amd
$ modprobe kvm_amd avic=1 nested=0

# dmesg | grep -i avic

kvm_amd: AVIC enabled
kvm_amd: x2AVIC enabled

# dmesg | grep -i vapic

AMD-Vi: Extended features (0x25bf732fa2295afe, 0x1d): PPR X2APIC NX GT [5] IA GA PC GA_vAPIC
AMD-Vi: Extended features (0x25bf732fa2295afe, 0x1d): PPR X2APIC NX GT [5] IA GA PC GA_vAPIC
AMD-Vi: Extended features (0x25bf732fa2295afe, 0x1d): PPR X2APIC NX GT [5] IA GA PC GA_vAPIC

qemu-kvm …. -machine q35,kernel_irqchip=split…..-global kvm-pit.lost_tick_policy=discard
```

On the guest VM, verify that the guest Local APIC is enabled in xAPIC mode by reading `MSR0x1b[11:10]`.

```
$ modprobe msr
$ rdmsr 0x1b --bitfield 11:10     # Should read "2"
 MSR0x1b[11:10]=2                   # Indicates LAPIC is enabled in xAPIC mode.
```

*Note: Guests with AVIC support up to 255 vcpus (8-bit APIC ID).*

```
$ modprobe msr
$ rdmsr 0x1b --bitfield 11:10    # should read "3" or 0b11
MSR0x1b[11]=1                    # indicates LAPIC is enabled in xAPIC mode
MSR0x1b[10]=1                    # indicates LAPIC is enabled in x2APIC mode
```

To verify x2AVIC:

1. Boot to OS, and then verify AVIC that support is present.

```
   <host># rdmsr -p 0 0xc00110dd
   MSR0xC00110DD.SVMRevFeatID[18].x2AVIC=1 indicates x2AVIC support
```

2. Edit and reload the `kvm_amd` module with AVIC enabled for the current session.

```
   <host># modprobe -r kvm_amd
   <host># modprobe kvm_amd avic=1 nested=0
```

3. Verify that kernel x2AVIC is enabled. The following command should return "1" & "0," respectively:

```
<host># cat /sys/module/kvm_amd/parameters/avic   -> 1 (or Y)
<host># cat /sys/module/kvm_amd/parameters/nested -> 0
```

4. To enable the VM to use the x2AVIC feature, edit its domain XML file as follows:

```
<host># virsh edit <domName>

<clock offset='utc'>
    ...
    <timer name='pit' tickpolicy='discard'/>
    ...
</clock>
```

In practice, you want the `qemu` command line to have the following configuration:

```
-M q35,kernel_irqchip=split
-global kvm-pit.lost_tick_policy=discard
```

5. Launch the guest VM, then verify that the guest Local APIC is enabled in x2APIC mode by reading `MSR0x1b[11:10]`.

```
<guest># rdmsr 0x1b --bitfield 11:10   (should read "3" or 0b11)
         MSR0x1b[11]=1 indicates LAPIC is enabled in xAPIC mode
         MSR0x1b[10]=1 indicates LAPIC is enabled in x2APIC mode
```

6. In the guest VM, run some high interrupt workload:

```
<guest># stress-ng --timer 32 --timer-freq 1000000
```

7. Next, run the following Linux performance utility to verify the VMEXIT counter:

```
<host># kvm_stat
```

If x2AVIC is active on the guest VM, then you should see `avic_incomplete_ipi` and `avic_unaccelerated_ entries` in the VM-EXIT column.

THIS PAGE INTENTIONALLY LEFT BLANK.

# Red Hat Enterprise Linux® Tuning Guide for AMD EPYC™ 9005 Processors

PID: 58469

Sandeep Gupta & Sylvester Rajasekaran