Performance Troubleshooting Case Studies for CloudOps Engineers

10 Real-world Linux Performance Scenarios with Commands, Tool Outputs, and Explanations

Case 1: High CPU Usage from Runaway Process

Problem: Application performance degraded and CPU utilization hit 100%.
Investigation:

```
$ top -b -n1 | head -15
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
2453 app 20 0 430m 25m 12m R 99.9 1.2 45:21 java
$ ps -p 2453 -o cmd=
/usr/bin/java -jar api-service.jar
```

Root Cause: A runaway Java thread consumed 100% CPU due to infinite loop.
Resolution: Restarted the application and implemented thread timeout logic.
Prevention: Enable APM profiling and add CPU usage alerting with Prometheus.

Case 2: Memory Leak Causing Swap Exhaustion

Problem: System became unresponsive after hours of uptime.
Investigation:

```
$ free -m
Mem: 16384 16200 184 Swap: 8192 8190 2
$ vmstat 2 5
si so high values indicating heavy swapping
$ ps aux --sort=-%mem | head
appuser 2132 92.0 ... python3 app.py
```

Root Cause: Memory leak in Python application gradually exhausted RAM and swap.
Resolution: Restarted leaking process and fixed code to release unused references.
Prevention: Implement memory profiling and restart policy for long-lived apps.

Case 3: I/O Bottleneck on Database Volume

Problem: Database transactions slowed drastically during peak hours.
Investigation:

```
$ iostat -xm 2 3
Device: rrqm/s wrqm/s r/s w/s await svctm %util
xvdb 0.00 1.00 30.0 200.0 210.3 12.0 99.8
```

Root Cause: Disk I/O saturation on database volume caused high wait times.
Resolution: Migrated database volume to provisioned IOPS SSD storage.
Prevention: Benchmark I/O patterns before production deployment.

Case 4: High Load Average but Low CPU Utilization

Problem: Load average exceeded 15 while CPU usage stayed low.
Investigation:

```
$ uptime
load average: 15.24, 14.95, 13.88
$ iostat -c 2 5
CPU idle >90%
$ ps -eLf | wc -l
Excessive blocked threads waiting for I/O
```

Root Cause: High load caused by blocked I/O threads, not CPU contention.
Resolution: Resolved disk bottleneck and optimized database queries.
Prevention: Correlate load average with process states, not CPU usage alone.

Case 5: Context Switching Due to Too Many Threads

Problem: Performance degraded due to kernel overhead handling thousands of threads.
Investigation:

```
$ vmstat 1 5
cs column > 300000 indicating excessive context switching
$ pidstat -w 1
Tasks: ctx/switch high for web-service PID 2334
```

Root Cause: Application created excessive worker threads leading to CPU thrash.
Resolution: Reduced thread pool size and optimized async I/O handling.
Prevention: Use connection pooling and asynchronous frameworks.

Case 6: CPU Throttling on Cloud Instance

Problem: Intermittent latency spikes observed without CPU saturation.
Investigation:

```
$ dmesg | grep throttled
CPU0: Package temperature above threshold, cpu clock throttled
$ cpupower frequency-info
current policy: frequency capped at 2.0 GHz
```

Root Cause: Thermal throttling reduced CPU performance under sustained load.
Resolution: Moved workload to higher-performance instance type with adequate cooling.
Prevention: Monitor thermal metrics and enforce cooling thresholds in cloud policy.

Case 7: Disk Queue Buildup During Backup

Problem: System slowdowns coincided with nightly backup window.
Investigation:

```
$ iostat -x 1 3
avgqu-sz 50.0 await 400ms %util 100.0
$ sar -d 1 5 | grep xvda
High I/O queue length during backup period
```

Root Cause: Backup jobs saturated disk I/O queue, starving other workloads.
Resolution: Rescheduled backups to low-load periods and enabled I/O throttling.
Prevention: Use ionice for background tasks and stagger heavy jobs.

Case 8: Network Saturation from Misconfigured NIC Queue

Problem: Throughput dropped on 10Gbps NIC under load.
Investigation:

```
$ ethtool -S eth0 | grep -i drop
rx_queue_0_drop: 10500
$ ethtool -g eth0
RX: 256 Current: 128
```

Root Cause: NIC ring buffer too small for high packet rate.
Resolution: Increased RX/TX ring buffers using ethtool -G and tuned IRQ affinity.
Prevention: Tune NIC buffers and enable RSS for high-speed interfaces.

Case 9: Performance Drop After Kernel Upgrade

Problem: Post-upgrade, users noticed sluggish performance across services.
Investigation:

```
$ uname -r
5.14.0-400.el9.x86_64
$ dmesg | grep 'sched'
Scheduler: CFS tuning changed
```

Root Cause: Kernel scheduler changes caused less efficient CPU distribution.
Resolution: Adjusted sysctl scheduler tunables and reverted to stable kernel.
Prevention: Test kernel versions in staging before production rollout.

Case 10: NUMA Imbalance on Large Multi-Core Instance

Problem: Multi-threaded service experienced uneven CPU utilization.
Investigation:

```
$ numactl --hardware
node0 cpus: 0-15, node1 cpus: 16-31
$ numastat
Huge imbalance in memory allocation across nodes
```

Root Cause: NUMA-unaware process allocated all memory on a single node.
Resolution: Restarted process with `numactl --interleave=all` for balanced allocation.
Prevention: Enable NUMA balancing and pin workloads appropriately.