Distinguish a container memory limit from a VPS out-of-memory event

An application can run out of memory while the VPS still has free RAM. A container, service or parent control group may impose a smaller budget. Conversely, an abruptly stopped VPS cannot usually explain a hypervisor incident through its own logs alone. Use evidence from the correct layer before deciding what needs more memory.

Start with the guest and the failed workload

1. Record the application failure time and whether the whole VPS rebooted.
2. Check current guest memory and boot time:

```
free -h
uptime -s
sudo journalctl -k -b --since "30 minutes ago" --no-pager
```

Look for memory-related kill messages around that time. Current free memory after a process exited is not a measurement of the pressure immediately before it exited. A Java heap error is also not the same event as the kernel killing a process.

Read the container configuration

For a Docker-managed workload, replace `my-container` with its name:

```
docker inspect --format 'Memory={{.HostConfig.Memory}} OOMKilled={{.State.OOMKilled}}
ExitCode={{.State.ExitCode}}' my-container
docker stats --no-stream my-container
```

The configured memory value is in bytes; zero means Docker did not set that specific hard limit. It does not prove that an ancestor cgroup is unlimited. The OOM flag and exit code are clues to combine with timestamps and logs. Exit code 137 by itself only indicates a SIGKILL-style exit, not its cause. Docker access is highly privileged; use it only on a server you administer.

Inspect cgroup v2 events when available

For a systemd service, obtain its control-group path:

```
systemctl show myapp.service -p ControlGroup
```

Replace `myapp.service`. On a cgroup v2 host, read the returned directory beneath `/sys/fs/cgroup`, using its actual path:

```
cat /sys/fs/cgroup/ACTUAL-GROUP/memory.current
cat /sys/fs/cgroup/ACTUAL-GROUP/memory.max
cat /sys/fs/cgroup/ACTUAL-GROUP/memory.events
```

`memory.max` may contain `max`. Event counters are cumulative; compare snapshots around a reproduced failure rather than treating an old nonzero count as a new incident. Ancestor limits matter. If these files are absent, check the cgroup version instead of creating files or changing kernel settings.

Decide what the evidence supports

A rising workload OOM counter alongside a finite limit supports investigating that workload’s budget. Guest-wide OOM logs need a broader memory review. A completely unresponsive or restarted VPS without a guest explanation needs provider-side investigation. Send support the service identifier, UTC failure time, relevant counters and redacted log lines. Do not disable the OOM killer or raise every limit to unlimited. After an agreed adjustment, repeat the workload and compare memory use and new event counts over a representative period.

Related CloudExa guides

- Check guest memory pressure
- Inspect application failures

Technical references

- Docker resource constraints
- Kernel cgroup v2 memory interface

Revision #1
Created 2026-11-10 10:51:41 UTC by CloudExa Hosting Admin
Updated 2026-11-10 10:51:41 UTC by CloudExa Hosting Admin