GRIDINSOFT
HELP CENTER

OFFICIAL SUPPORT GUIDE

Z

Zombie Process: Meaning, Detection, Cleanup, and Prevention

A zombie is an exited child whose parent has not collected its status. It uses no CPU, cannot be killed again, and is fixed through the parent process.

PUBLISHER	UPDATED	READ TIME
Gridinsoft Support	Oct 8, 2026	3 min

A **zombie process**, also called a **defunct process**, is a child process that has already terminated but whose parent has not yet collected its exit status. The kernel retains a small process-table entry so the parent can call `wait()`, `waitpid()`, or a related function and learn how the child ended.

A zombie is not still executing. It uses no CPU and has already released almost all memory and open resources. It retains a PID and minimal accounting information. This behavior is documented in the Linux wait(2) manual.

Zombie vs. orphan or stuck process

- **Zombie:** finished but not reaped; usually shown with state `Z` or `<defunct>`.

- **Orphan:** still-running child whose original parent exited; it is adopted by init, systemd, or a subreaper.
- **Sleeping process:** alive and waiting for an event; it is not a zombie.
- **Uninterruptible process:** often shown as `D` on Linux and waiting on kernel I/O. Its cleanup is a different problem.

Why zombie processes appear

A short-lived zombie state is normal between a child's exit and the parent's next wait call. Persistent or accumulating zombies usually mean the parent application is not reaping children correctly, is stuck, or has a bug in its signal-handling logic. Containers also need a suitable PID 1 process that forwards signals and reaps children.

How to find zombie processes

On Linux, inspect process state and parent PID with:

```
ps -eo stat,pid,ppid,etime,comm
```

Look for a state beginning with `Z`. Record the zombie PID, its PPID, elapsed time, and command. You can inspect one parent with:

```
ps -p PPID -o pid,ppid,user,stat,etime,cmd
```

Tools such as `top` may show an overall zombie count, but a count alone does not identify the faulty parent.

Can you kill a zombie process?

No. Signals such as `SIGTERM` or `SIGKILL` cannot terminate it again because it is already dead. The parent must reap it. Randomly using `kill -9` on PIDs is ineffective and can disrupt an unrelated service if a PID has been reused.

Safe cleanup order

1. **Assess impact.** One recent zombie is normally harmless. Investigate when the count grows, persists, or approaches process limits.
2. **Identify the parent.** Check what service or container owns the PPID and whether it is healthy.
3. **Use the application's normal controls.** Update or gracefully restart the faulty service during an appropriate window. Do not stop an important database or system service without understanding the impact.
4. **Escalate carefully.** If the parent cannot recover, follow the service's documented restart procedure. When the parent exits, init or a subreaper normally adopts and reaps the zombie.
5. **Fix the cause.** Preserve logs, reproduce the leak, update the application, and report the bug instead of relying on repeated reboots.

Developer prevention

Parents should correctly handle all child exits using `wait()` or `waitpid()`, including multiple children per signal. Avoid doing unsafe work inside signal handlers. Supervisors and containers should use a proper init or subreaper when the main application does not reap descendants.

Zombie process FAQ

Is a zombie process malware?

No. It is a normal operating-system state. Malware can create processes, but the state itself is not evidence of infection. A zombie computer is a different concept: an attacker-controlled device.

Does a zombie consume RAM?

Its address space is gone. Only a small kernel table entry remains, but enough accumulated entries can exhaust available PIDs or process slots.

NEED MORE HELP?

We are here when the guide is not enough.

Visit the Gridinsoft Support Center for current product guidance or submit a support request with the details of your case.

Open this guide online at support.gridinsoft.com