Fix Linux inode exhaustion when disk space is still available

A server can report “No space left on device” while its storage chart still shows free gigabytes. One possible cause is inode exhaustion: the filesystem cannot create another file or directory even though it has room for more file contents. This guide isolates that condition from an ordinary full disk. It applies to GNU tools on Debian and Ubuntu; filesystem behaviour varies.

Check the exact destination

1. Copy the path from the application error. If the file does not exist, use its existing parent directory. Replace `/var/lib/myapp` below with that directory.
2. Compare block space and inode space on the same filesystem:

```
df -hT /var/lib/myapp
df -i /var/lib/myapp
findmnt -T /var/lib/myapp
```

Look at `IFree` and `IUse%`. A zero free-inode count is strong evidence for this cause. A full block-space result points to a different problem. Do not compare the root filesystem with a separate application volume and assume their available capacity is shared.

Find where the file count grew

Start inside the affected application directory, rather than scanning the entire server during peak traffic. GNU `du` can report inode counts:

```
sudo du --inodes -x --max-depth=1 /var/lib/myapp
```

Repeat the command for an unexpectedly large child directory. The `-x` option avoids traversing different filesystems. This is a read-only scan, but millions of directory entries can still make it slow. Cancel it if it worsens an active incident. Check the application documentation before interpreting a large cache, session, mail queue or build directory as disposable.

Choose a controlled cleanup

Identify the process producing the files and its retention setting. A cleanup that removes yesterday's files while a broken job generates thousands more each minute will only provide temporary relief. Prefer the application's own expiry or queue-management command after confirming what it removes. Keep required backups and do not recursively delete an unfamiliar directory.

Moving files into another folder on the same filesystem does not free their inodes. Creating a local archive may also fail when no inode remains, and the archive itself needs block space. Plan any export to a separate verified destination before modifying the originals.

Verify and ask for help

After an approved cleanup, repeat both `df` checks and retry the failed application action once. Record whether inode use starts climbing again. If inode capacity is available but the error persists, report the exact path, filesystem type, application error and timestamp to support; quotas, container limits or another mount may be involved. Never format the disk or rebuild the filesystem merely to clear this symptom.

Related CloudExa guides

- Find large files and directories
- Check the intended mounted volume

Technical references

- GNU df manual
- GNU du manual

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