Verify a Linux backup download before you rely on it

A successful download gives you a file. A recovery rehearsal tells you whether that file contains the data and configuration you need. Use both checks before deleting an older backup or making a risky application change.

Record the source and expected contents

This guide uses GNU tools on a Linux VPS or a Linux workstation. It does not require changes to your live application. Record the service, backup name, creation time and expected contents. Note any excluded folders and whether a separate database export belongs with the archive. Keep credentials and recovery keys private.

Finish the download before hashing it. Compare the reported size where the source provides one, but remember that two different files can have the same size. If a backup service offers a SHA-256 checksum through a trusted interface, save that value alongside your download. If you control the source machine, calculate a checksum there before transfer.

Compare the checksum

For an archive you created and control, change to its directory and generate a manifest before copying both files to independent storage:

```
sha256sum backup.tar.gz > backup.tar.gz.sha256
```

After transfer, run this in the destination directory:

```
sha256sum --check backup.tar.gz.sha256
```

Expect an OK result for the exact filename. GNU's SHA-2 tools compare file contents against the saved digest. A mismatch can indicate an incomplete transfer, the wrong file or corruption. Keep the original and investigate rather than editing the checksum to match.

A checksum you generate only after downloading cannot prove that the transfer matched the source. It provides a baseline for detecting later changes. Likewise, a checksum stored beside a maliciously replaced archive does not establish who created it. Obtain trusted backup files and trusted reference values.

Inspect without extracting into production

For a trusted gzip-compressed tar archive, list the entries first:

```
tar -tvzf backup.tar.gz
```

Check for the expected world or application directory and configuration files. Watch for absolute paths, parent-directory traversal entries and unexpected symbolic links. Do not extract an untrusted archive as root. Large archives can take time to list; allow the command to complete instead of launching repeated copies.

Use a separate disposable environment for recovery, with enough free space for the expanded data. Follow the archive tool's security guidance and inspect permissions and links before starting applications. Never choose the live application directory as an extraction target for a rehearsal.

Prove that the application can recover

Restore the matching database into its own isolated test instance when required. Prevent the test from sending production emails, charging payments, running public webhooks or joining a live game network. Use the same application and runtime versions as the recovery point unless you are testing a documented upgrade.

Open representative data: join the recovered game world, inspect a sample inventory, or read a known application record. Record the result, duration and missing dependencies. A readable

archive still may omit a database, encryption key or uploaded-file directory.

Keep the original backup unchanged while you investigate a failed rehearsal. Cleanup should affect only the disposable test environment after you have recorded the result. Follow our Linux disk-space guide to measure storage first, and use the support-ticket checklist if a CloudExa backup cannot download or reports a checksum error.

Revision #1
Created 2026-11-09 09:49:46 UTC by CloudExa Hosting Admin
Updated 2026-11-09 09:49:46 UTC by CloudExa Hosting Admin