Diagnose Rust custom map checksum mismatches without deleting the world

A “World File Mismatch” or map checksum error usually needs an investigation of which map bytes the server and client received. Deleting the world is not a diagnostic step. Begin with the service stopped if players cannot join, and preserve the current files before experimenting with names or URLs.

Capture the mismatch

Record the exact client error, time, Rust version and affected connection address. Ask whether everyone is affected or just players who downloaded an earlier map. Save the server's startup output showing the map name, and note the effective `server.levelurl`. Keep access tokens out of tickets; a public map URL should not contain administrator credentials.

Take a backup of the identity folder, current map and startup configuration. Keep the original downloaded map separately. Do not rename a saved world to force a different map to accept it, and do not replace the production map with a file sent by an unknown player.

Compare the delivery chain

1. Open the configured URL outside your file-host account. Confirm the response is the actual map. Facepunch notes that an unrelated HTTP page can be returned where Rust expects map content.
2. Download the file locally and compare its byte size and checksum with your authorised release copy. Record the algorithm used so staff compare like with like.
3. Compare that release with the map present on the server. Use an authorised download through the game panel; do not edit it in place.
4. Review recent changes. Was a different map uploaded under the same filename? Was an existing map renamed? Did the URL start redirecting to a login page?

5. Check the author's release requirements against the installed Rust build. A matching filename alone does not establish matching content or compatibility.

Choose the smallest safe repair

If the URL returns a web page, repair the public download at the file host and recheck it before changing the server. If the map release changed, use an explicit new versioned file and an isolated test instance. The hosting instructions explain why players need a direct, continuously available download.

Facepunch warns that custom maps are not downloaded afresh on every start when already present, and that mid-wipe terrain changes can leave existing entities in inappropriate positions. Therefore, do not follow a cache-deletion recipe against a live world without a reviewed recovery plan. Ask support and the map author to agree the exact files involved, expected save compatibility and maintenance window.

Verify and recover

Test the corrected release on a separate identity and have both a previously affected player and a fresh downloader join. Record which map version each received. Confirm terrain and representative buildings before returning production to service.

If checks fail, stop the test and retain the original production map and save set. Restore the original URL and configuration only with the corresponding unchanged files. Keep evidence of the download response and checksums for support. This guide intentionally avoids renaming protocol-version saves or deleting player caches as a blanket fix; those actions can hide the actual distribution error.

Related help

- Find Rust server.cfg and the correct identity save folder
- Schedule game server backups and check retention
- Diagnose Rust game, query and RCON connection ports

Revision #1
Created 2026-11-09 10:42:41 UTC by CloudExa Hosting Admin
Updated 2026-11-09 10:42:41 UTC by CloudExa Hosting Admin