Preflight a Rust custom map download before the next wipe

A custom map can work for the owner and still fail for every joining player if its URL requires a login or returns a download page. Check the delivery path before your announced wipe. This procedure prepares a future map; it does not authorise replacing the current world during an active wipe.

Keep the current release recoverable

Save the existing startup settings, map URL, map file and current backup identifier. Choose a maintenance date with staff and decide which player data your announced wipe policy preserves. Keep the old download available while the new release is being checked. Use an isolated test service with its own identity and allocations; ask support if you do not have one.

Obtain the map from its authorised author and record the exact release filename. Check the author's Rust-version requirements, distribution permission and any required server extensions. Do not upload a paid map to a publicly indexed repository unless its licence explicitly allows that delivery method.

Check the URL as a player would

1. Publish the map at a stable public HTTPS address under your control or an authorised download service. Use a versioned filename rather than replacing unrelated content behind a familiar name.
2. Open the address in a private browser session with no provider login. Confirm it returns the map download itself, not an HTML page, advertising interstitial or sign-in screen.
3. Download a second copy from another network if practical. Compare its size and a locally calculated checksum with the author's file. Matching checksums verify identical bytes, not

that the map is safe or compatible.
4. Remove short-lived access tokens from the delivery design. Players may need the file long after the administrator's initial test.
5. Check any bandwidth or download limits with the file host. A successful single download does not demonstrate capacity for an entire wipe launch.

Test without touching production

Facepunch's custom-map instructions use `server.levelurl` for the direct download and advise removing procedural seed and world-size arguments for this configuration. Apply those changes only to the test service first, leaving its unique identity and assigned ports in place. A local `file:///` reference is not a substitute for the URL that remote players need.

Start the test instance and read its map-loading output. Join from a normal Rust client using the matching supported game version. Check terrain, spawn and access to a few key monuments. Record any author-recommended tests and the result rather than simply writing “map works.” Stop the test if it tries to load a production identity.

Release or roll back

Only after the test and announced wipe window should you apply the approved URL to production through the supported startup editor. Take the agreed checkpoint first and watch the first start. If the map cannot be downloaded, stop and restore the previous startup settings before further changes; do not delete the current saves to make startup continue.

Map compatibility and server extensions change between Rust releases. Repeat the preflight for each map revision. Keep the exact map file, URL and author release notes with your backup record so a later restore is not dependent on remembering which “final” upload was used.

Related help

- Find Rust server.cfg and the correct identity save folder
- Schedule game server backups and check retention

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