Update Assetto Corsa content and isolate extension conflicts

When a race stops working after a content update, identify whether the change affected the car, track, server software or an extension. Keep those changes separate during testing. This guide covers original Assetto Corsa and explicitly distinguishes the stock dedicated server from the optional AssettoServer implementation.

Preserve the working combination

Record the server implementation and version, car and track releases, selected layout, active preset and any client requirements stated by the content author. Save a copy of the entire working content and configuration set. Include results and extension data where present, then verify the backup can be opened.

Stop the service normally before replacing active files. Keep an untouched working copy and a candidate copy. If there is no staging service, arrange a maintenance window and retain a clear return path instead of experimenting while players are racing.

Classify the first failure

1. **Server fails before players join:** capture the earliest startup error and the exact files changed. Check archive layout, folder identifiers and the selected track layout before altering network settings.
2. **Only updated clients fail:** compare the actual car or track release with the server's release. A shared display name does not establish identical content. Use the author's original package rather than collecting individual replacement files from different versions.
3. **A particular car fails:** compare its configured model in the entry list with the allowed car set. Recheck layout pit capacity when changing the entry list.

4. **Driving works but an extension fails:** investigate that extension separately. Retain its error log and configuration rather than replacing the race server or clearing all content.

Apply the correct extension rules

For the stock `./acServer` service, use the Kunos configuration reference and the specific extension author's instructions. AssettoServer's `extra_cfg.yml` and bundled-plugin configuration do not control the stock implementation.

For the separate `./AssettoServer` service, the upstream update guide warns that extra and plugin configuration parameters can change between releases. Prepare the new release in staging and transfer intended settings into its current generated configuration rather than blindly replacing it with an old file. Preserve any plugin-specific database or credentials separately.

AssettoServer's missing car checksum diagnostic identifies missing car data. Repair the complete content package instead of suppressing protections. A YAML parsing error is another distinct case: inspect the changed indentation and spelling against the current generated configuration, not a copied example for another version.

Verify and retain a rollback

Change one content release or extension at a time in staging. Test a real join, car selection and several minutes of driving, then test the extension's advertised feature. Repeat after a controlled restart so success does not depend on a previous process's state.

If the candidate fails, preserve its files and logs and restore the stopped service's complete working set. Tell players which exact content versions to use. Avoid combining an old server preset with a newly updated track or mixing extension data from different releases without the author's migration guidance.

Related CloudExa guides

- Back up game files before making changes
- Diagnose Assetto Corsa startup, lobby and join failures

Revision #1
Created 2026-11-10 11:05:47 UTC by CloudExa Hosting Admin
Updated 2026-11-10 11:05:47 UTC by CloudExa Hosting Admin