Introduce CMI without conflicting with existing server plugins

CMI combines many server-management features in one premium plugin. That convenience also creates overlap with existing chat, homes, teleport and economy plugins. A careful introduction enables one feature at a time instead of replacing several working systems in a single restart.

Prepare the correct files

Obtain CMI from an authorised source using a valid licence and check the author's supported Minecraft and Java requirements for your release. Install the required CMILib version too. Do not use redistributed or modified premium jars; they can differ from the documented build and put your server credentials at risk.

Stop the server and take a completed backup of the plugin directory, worlds and external databases before installation. Preserve the current plugin list and configuration. Use the plugin installation workflow to avoid duplicate jars or editing files while the server writes them.

Start to generate configuration, read Console for errors and stop again before editing. Follow the files generated by your installed build. The online command reference includes historical version notes, so compare its instructions with the help available in your release.

Assign ownership of overlapping features

Write a short list of the features already provided by other plugins: economy balances, chat formatting, homes, warps, teleport requests and kits. Choose one owner for each. Leaving two plugins responsible for the same command or event can produce inconsistent results even when both appear enabled.

Use `/cmi checkcommand home` to find CMI commands related to home, following the installed help for its output. Check aliases and namespaced routes before removing or replacing anything. Do not assume typing a familiar command proves which plugin processed it.

Leave economy migration out of the initial trial. CMI's economy has specific configuration and Vault integration requirements documented by the author. Installing a random Vault jar is not proof that every economy plugin can read CMI balances. Preserve the current economy provider until a separate migration has been planned and tested.

Enable a small feature set

Select one low-impact feature you actually need and read its permission entry. CMI generally documents command permissions under `cmi.command` and command-specific nodes, with additional permissions for actions involving other players. Give the test group only the nodes needed for your chosen action.

Use a non-operator account and check both allowed and refused actions. A player should not gain staff capabilities merely because a basic convenience command works. Use LuckPerms group testing to inspect inherited permissions rather than adding a broad wildcard when something is denied.

Check persistence and integrations

Restart normally and repeat the feature test. Review Console for missing-library errors, command conflicts or storage failures. If you later trial an economy integration, compare the same disposable account through the balance display and the consuming shop plugin, using a small controlled transaction.

Do not run a bulk import, reset or conversion against live player data while exploring menus. Test migration tools against an isolated copy with production database access removed, and record how existing homes, balances and identifiers will map.

Get useful support evidence

Supply the CMI, CMILib, server and Java versions; the exact command; the expected behaviour; and a redacted error excerpt. Remove passwords and licence details from screenshots. Further reading: CMI documentation, command reference and CMILib.

Looking for another plugin? Browse the Minecraft plugin guide directory.

Revision #2
Created 2026-11-09 08:57:17 UTC by CloudExa Hosting Admin
Updated 2026-11-09 08:57:18 UTC by CloudExa Hosting Admin