Multiverse-Core tasks

Task guides and troubleshooting for Multiverse-Core. Check the installed release before changing configuration or data.

- Set a Multiverse world spawn without changing the main server world
- Find why a player can teleport to one Multiverse world but not another
- Choose a Multiverse respawn world and test bed behaviour separately
- Stop weather changes in one Multiverse event world
- Diagnose Multiverse animal and monster spawn rules in one world
- Apply a world-specific PvP rule without bypassing region protection
- Change difficulty for one Multiverse world and confirm it persists
- Use a Multiverse world alias without renaming the world folder
- Prevent an unused Multiverse event world from loading automatically
- Investigate a Multiverse world player limit that staff can bypass

Set a Multiverse world spawn without changing the main server world

A world-specific spawn can differ from the main server spawn, a player bed and a first-join plugin destination. Record which arrival path you intend to change before moving the point, so you can test that path without disturbing other worlds.

Work through the change

1. Identify the exact managed world name from Multiverse information. Stand on a safe solid platform with room above the player and record its coordinates and facing.
2. Check the set-spawn command in the installed Multiverse command help. Set the spawn for that named world rather than changing server.properties or the server main-world name.
3. Use the documented world-spawn teleport from an ordinary test account. Record where the player lands and whether Multiverse adjusts the target to a nearby safe location.
4. Test a second arrival method, such as the portal that players normally use. If it has an explicit coordinate destination, inspect that destination separately instead of repeatedly moving the world spawn.

Check the result

The world-spawn route should reach the platform with the intended facing. A bed respawn or EssentialsX first-join action may still choose another location. Compare these routes independently and keep the results with the world name so staff do not mistake a different arrival rule for a failed save.

Check whether the login plugin or a first-join plugin teleports players after Multiverse selects a spawn. Those later actions can make a correct world spawn appear ineffective. Compare an established player’s world teleport with a new account’s login before changing another spawn setting.

Recover or stop safely

Return to the recorded old spawn and restore it with the same world-specific control if the new location is unsuitable. Preserve the platform until players have tested it. Avoid deleting and reimporting a world merely to update a spawn point.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Find why a player can teleport to one Multiverse world but not another

A world access denial can come from the target world, the teleport command or a portal permission. Compare those layers with one ordinary player account. Using an operator account hides the permission path that your players actually need.

Work through the change

1. Record the source world, target world and exact command or portal used. Check that the target is loaded and listed under the expected name rather than only an alias.
2. Read the installed Multiverse access enforcement setting and the documented target-world access permission. Inspect the player’s inherited and temporary grants without adding a wildcard.
3. Test a permitted world through the same command or portal. Then test the denied world and record both responses, including any protection or permission plugin messages.
4. Grant only the missing documented capability to a test role if your policy permits access. Reconnect that account and repeat the original route before changing permissions for the full player group.

Check the result

The approved player should reach the target through the intended route while a control account remains denied. If a direct teleport works but the portal fails, inspect the portal-specific permission instead of expanding world access again. Record the exact grant that changed the result.

Write down the exact world name used by the permission node and the command. A display alias can look identical to the intended destination while referring to another internal name. Test denial from a second source world if players normally arrive through more than one route.

Recover or stop safely

Remove the temporary grant if the route remains broken or grants extra access. Restore the previous role assignment and keep the denial messages for support. An unloaded or renamed destination needs a world-reference repair, not a broad administrator permission.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Choose a Multiverse respawn world and test bed behaviour separately

Players can follow different respawn rules after death, a bed use or an End exit. Set a deliberate destination for one source world and test those cases with expendable inventory before announcing a server-wide respawn policy.

Work through the change

1. Record the source world’s respawn-world and bed-respawn settings along with its normal spawn coordinates. Keep the intended target world loaded and confirm its landing point is safe.
2. Use the world-property control documented for your Multiverse release to set the target respawn world. Keep bed behaviour unchanged during this first trial so you change one cause at a time.
3. Test a controlled death with an ordinary account that has no valuable items. Compare an account with a valid bed to one without a bed, and note the arrival world in each case.
4. If the source is an End world, test its exit route separately. Inspect other spawn or inventory plugins when their policies redirect the player after Multiverse chooses a destination.

Check the result

The no-bed test should follow the intended world rule, and the bed test should follow your documented bed policy. Confirm hunger, inventory and game mode after arrival when other multi-world plugins manage them. A correct coordinate in the wrong world is still a failed respawn test.

Use separate trial deaths for a valid bed, an invalidated bed and no bed. Record the entry world and final world each time. Testing only the default death case leaves the common complaint unresolved when a player’s saved bed or anchor changes the selected destination.

Recover or stop safely

Restore the earlier respawn-world property and any bed setting you changed. Return the test account to its starting location and preserve its original player data. Do not erase bed records or reset a whole world to investigate a respawn conflict.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Stop weather changes in one Multiverse event world

An event map can need stable weather while survival worlds keep normal conditions. Apply the policy to the named event world and check its persisted property, rather than running an unspecific weather command from whichever world an administrator happens to occupy.

Work through the change

1. Record the event world name and its current weather property. Check scheduled commands or event plugins that may also change rain and storms during the same period.
2. Read the weather property documentation for your installed Multiverse release. Apply the intended world-specific weather policy and record the value returned by its information command.
3. Observe the event world over an interval that normally allows a weather change. Check a separate survival world during the same interval to confirm you did not apply a global rule.
4. Return after a normal planned server restart to verify persistence. If a scheduler changes the weather, disable only that conflicting test action or adjust its target after recording its previous behaviour.

Check the result

The event world should keep the chosen conditions while the control world follows its existing policy. Client weather effects, region settings and visual packs can differ from actual server weather, so verify from more than one position and compare the world information output.

Observe a full weather transition rather than checking only the immediate sky. Another weather plugin may reapply its schedule later. Record whether the goal is clear weather, rain or a frozen current state, because those are different policies with different checks after a restart.

Recover or stop safely

Restore the original weather property when the event ends and remove any temporary schedule override. Keep the earlier values in the event checklist so another staff member can return the world to normal without guessing which command produced the temporary result.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Diagnose Multiverse animal and monster spawn rules in one world

A quiet world may reflect a configured spawn category, a gamerule, light conditions or a mob cap. Begin with the world’s entity spawn configuration and a controlled natural-spawn test rather than changing every server-wide mob setting.

Work through the change

1. Identify the world and record its difficulty, entity spawn categories and exceptions. Read the current Multiverse entity-spawn-config documentation because older animals and monsters examples may use different controls.
2. Choose one category and one safe observation area. Check time, light and player distance so the trial does not depend on conditions that already prevent normal natural spawning.
3. Change only the documented category property required for the trial. Leave exception lists intact until you have recorded which species they permit or exclude.
4. Observe natural spawns over a suitable interval and compare another world. Keep command-spawned and plugin-spawned mobs out of the result, since they may follow different event paths.

Check the result

The test should show the chosen natural-spawn policy without changing the control world or existing animals. A successful configuration message does not prove a species has room under the cap. Inspect competing plugins and actual spawn conditions if the result remains empty.

Test an actual new spawn attempt separately from already-present mobs. Existing entities can remain after future spawning is restricted. Keep the test in appropriate light and biome conditions so normal Minecraft spawn requirements do not disguise whether the world rule is functioning.

Recover or stop safely

Restore the original category value and exceptions if the trial creates unwanted activity. Move players away while you assess extra entities; do not use a blanket world purge that could remove pets, named mobs or plugin-managed creatures along with the test population.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Apply a world-specific PvP rule without bypassing region protection

World PvP and region PvP can both influence a fight. Check the broader Multiverse property and a small unprotected test area before altering spawn regions. A permission bypass on staff accounts can conceal the policy that ordinary players experience.

Work through the change

1. Record the target world’s PvP property and any nearby region or claim rules. Choose a test area where both players can stand safely without valuable equipment.
2. Use the installed world-property command to change PvP only for the named world. Retain the previous value and avoid touching unrelated combat or invulnerability settings.
3. With two ordinary accounts, try one controlled hit in the test area. Then repeat in the protected spawn without changing its flags, noting whether each hit applies damage.
4. Test another world that should keep its original policy. If results differ between identical positions, inspect staff bypass, team rules or a combat plugin before broadening any permission.

Check the result

The world policy should behave as intended in the control area while existing region rules continue protecting their areas. Check projectile and melee behaviour separately if your gameplay depends on both. A test with a single operator cannot establish what two regular players can do.

Use disposable equipment and willing players for damage tests. Verify neighbouring regions too, because a global combat setting can resemble a local rule during a single encounter. Keep any invincibility or event protection in the notes so later staff do not misdiagnose the interaction.

Recover or stop safely

Restore the earlier Multiverse PvP value if the world-wide change affects an event or minigame. Keep region flags at their recorded values. Recover any test inventory from the prepared checkpoint rather than changing global death rules to make the trial convenient.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Change difficulty for one Multiverse world and confirm it persists

A difficulty change for an event or resource world should not silently alter the main survival world. Record the target world and the player impact first, especially when a peaceful setting could remove hostile entities that are important to an existing encounter.

Work through the change

1. List the world through Multiverse and record its current difficulty. Identify any saved or plugin-managed mobs that require a staged copy before trying a different difficulty.
2. Choose the named world-property control from your installed documentation. Apply one intended difficulty value and read back the value from world information.
3. Test with a normal account in that world and inspect the relevant gameplay effect. Check the main world separately rather than assuming a command context selected the right target.
4. After the next planned restart, compare the persisted difficulty with your notes. If it changes back, inspect scheduled commands and world management plugins before rewriting the property repeatedly.

Check the result

The target world should retain the chosen difficulty and the control world should retain its earlier one. Existing entities and combat plugins may not produce an immediate obvious demonstration, so use the reported property alongside a controlled gameplay test. Record the restart result for later staff changes.

Check the saved value after a normal restart rather than treating immediate feedback as persistence proof. If the difficulty returns to its former value, identify the configuration or plugin that reapplies it. Avoid repeatedly increasing difficulty while the underlying source of the overwrite remains unknown.

Recover or stop safely

Restore the original difficulty if the trial disrupts play. Restoring a difficulty value does not resurrect entities removed during a peaceful trial, so use the staging checkpoint when recovering those encounters. Avoid a live peaceful test on a world with irreplaceable hostile mobs.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Use a Multiverse world alias without renaming the world folder

An alias gives players a friendlier label while the world’s internal identity remains stable. That separation can prevent broken portals and inventories, but commands and third-party plugins may resolve aliases differently. Test the display change before telling staff to use it everywhere.

Work through the change

1. Record the managed world name, folder name and existing alias. Identify portals, menus and commands that currently refer to the world so you can verify they still resolve it.
2. Use the alias property documented for the installed Multiverse version. Choose a label that is distinct from another world name and avoid formatting codes during the first trial.
3. Check the world list and any menu that displays the label. Test an existing teleport with the original internal name before trying the alias as a command argument.
4. Inspect dependent plugin settings without renaming their references by rote. Keep the real world name in maintenance notes and use the alias only where that plugin explicitly supports it.

Check the result

Players should see the new label in the intended display while existing destinations still reach the same saved world. An alias should not create a new folder, reset a seed or move inventories. A plugin that ignores the alias may need its own display setting rather than a world rename.

Keep the folder name and alias in separate columns of the change record. That distinction matters when another plugin expects an internal world identifier. Give players the friendly label while retaining the exact source name for portals, permission nodes and recovery notes.

Recover or stop safely

Restore the earlier alias if a label collision confuses a menu or command. Keep the world folder and internal name unchanged throughout the trial. If a dependent command fails, revert that one display edit and collect its exact argument instead of reimporting the world.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Prevent an unused Multiverse event world from loading automatically

An event world can consume resources even when no event is scheduled. Disabling its automatic load changes startup behaviour, not the contents of its folder. Audit every route to that world before leaving players with a portal that points at an unavailable destination.

Work through the change

1. List players, spawn rules and portals that refer to the event world. Move test accounts to a stable lobby and record the world’s current auto-load property.
2. Use the installed Multiverse property control to disable auto-load for that named world. Leave the world folder and its managed-world record intact for the planned return.
3. If you need to unload it now, use the documented world unload path only after checking players and dependent tasks. Record any refusal rather than forcing files away while the server is running.
4. At a planned restart, confirm the event world stays unloaded and the lobby loads normally. Disable or explain its public menu link until the next event so players do not interpret the closure as lost data.

Check the result

The startup world list should omit the closed event world without creating another world under a similar name. Retain a verified backup and the exact folder name. A plugin with its own world loader may still load it, so inspect that plugin’s startup behaviour if it reappears.

Ask the event owner whether any scheduled task, portal or backup routine expects the world to be loaded. Disabling automatic load changes availability, not merely resource consumption. Keep the documented manual return path with the event calendar so staff can prepare the map before participants arrive.

Recover or stop safely

Re-enable auto-load and use the documented load command when reopening the event. Verify portals and player arrival before advertising access. Do not delete the managed record or folder to achieve a temporary startup policy; those actions introduce unnecessary recovery work.

> CloudExa game panel file manager
> The CloudExa file manager can help locate plugin configuration and saved diagnostic files. This is the hosting panel, not the plugin’s own interface.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.

Investigate a Multiverse world player limit that staff can bypass

A world player limit can protect a small event, while staff bypass permissions let particular accounts enter. Check the limit and bypass separately. A working staff teleport does not establish that a normal participant should be admitted when the world is full.

Work through the change

1. Record the named world’s configured player limit and current occupancy. Check which staff or test accounts inherit a player-limit bypass, including world-specific and broad grants.
2. Create a small controlled admission trial with expendable accounts. Use the limit property from your installed Multiverse release and keep a route back to the lobby.
3. Fill the test capacity with ordinary accounts and attempt another ordinary admission. Then try a staff account with the documented limited bypass to compare the two responses.
4. Check reconnect behaviour and the event’s usual portal or menu route. If a separate queue or minigame plugin also enforces admission, keep its result distinct from Multiverse’s world count.

Check the result

Ordinary accounts should respect the intended capacity while authorised staff follow your bypass policy. Confirm that the configured world count is not being confused with your hosting plan’s server slots. Changing this property does not add RAM, CPU or a larger service allocation.

Use an agreed fallback lobby for capacity tests and check its own entry permission. A rejection message is not enough if the rejected player becomes trapped at a portal. Keep the trial limit small and restore normal admission promptly after the observation window.

Recover or stop safely

Restore the earlier world limit after the trial and remove temporary bypass grants. Return the test accounts to the lobby before closing access. If the event becomes overfull, use its documented evacuation procedure with a player notice rather than killing processes or deleting player files.

Related guidance and maintainer reference

For the broader setup, read Add and import Minecraft worlds with Multiverse-Core. Check the installed plugin version before applying syntax from the official Multiverse-Core documentation.