[security] fix(commands): keep bridge local-only by default #208

Merged **tjb-tech** merged 1 commit into `HKUDS:main` from `Hinotoi-agent:fix/remote-bridge-c…`

4 days ago

Hinotoi-agent commented 4 days ago • edited

Contributor

Summary

This PR hardens a residual remote-gateway slash-command boundary related to the already-public command hardening family addressed in #127.

The `/bridge` command can spawn bridge-managed shell sessions through `/bridge spawn CMD`. It was still registered with the default `remote_invocable=True`, which means an accepted remote channel/gateway sender could reach the command path from chat unless the deployment separately restricted the sender at the channel layer.

This patch makes `/bridge` local-only by default and adds regression coverage for both the command metadata and gateway enforcement path.

Security issues covered

Issue	Impact	Severity
Remote `/bridge spawn` shell execution	Accepted remote gateway users could execute OS commands through the bridge command path	High

Before this PR

- `SlashCommand.remote_invocable` defaults to `True`.
- `/bridge` was registered without overriding that default.

- The remote gateway command path allows commands unless `remote_invocable` is explicitly false.
- `/bridge spawn CMD` forwards attacker-controlled command text to the bridge session manager.
- The bridge session runner starts the command through the shared shell subprocess helper.

After this PR

- `/bridge` is registered with `remote_invocable=False`.
- `/bridge` is marked `remote_admin_opt_in=True`, preserving an explicit trusted-operator opt-in path consistent with the existing gateway model.
- Remote `/bridge spawn ...` messages are denied by the gateway before the command handler runs.
- Regression tests cover the command metadata and the remote gateway blocking behavior.

Why this matters

Remote channel/gateway users are a different trust boundary than a local OpenHarness UI operator.

`/bridge spawn` is not a read-only helper: it can start arbitrary shell commands in the configured project working directory as the OpenHarness process user. If reachable from remote chat, it can expose local files, credentials, workspace state, and repository contents, and it can mutate the working tree or start long-running processes without going through the normal model/tool permission flow.

How this differs from #127

#127 added the remote command boundary and fixed known sensitive commands such as `/permissions` and the `/memory show` traversal path.

This PR is a same-family residual fix, not a claim that the earlier patch was incomplete in scope:

- **fix: harden gateway slash command security** #127 introduced the `remote_invocable` and `remote_admin_opt_in` model.
- `/bridge` remained registered with the default remote-invocable behavior.
- The vulnerable sink here is different and stronger: `/bridge spawn` reaches shell subprocess creation, not permission-mode mutation or memory-file reads.
- The fix applies the existing **fix: harden gateway slash command security** #127 boundary model to a missed bridge control-plane command.

Both fixes are needed because the gateway can only block a sensitive command if that command is explicitly marked local-only.

Attack flow

```
Accepted remote channel/gateway user
    -> sends /bridge spawn CMD
```

```
          -> ohmo gateway parses it as a slash command
              -> command defaults to remote_invocable=True
                  -> _bridge_handler() accepts spawn
                      -> BridgeSessionManager.spawn(command=CMD)
                          -> create_shell_subprocess()
                              -> bash -lc CMD
```

Affected code

Area	Files
Command metadata	`src/openharness/commands/registry.py`
Gateway regression coverage	`tests/test_ohmo/test_gateway.py`
Registry regression coverage	`tests/test_commands/test_registry.py`

Root cause

- The default slash-command metadata is remote-allowed.
- `/bridge` was not explicitly marked local-only even though one of its subcommands starts shell processes.
- The remote gateway relies on command metadata to decide whether a slash command can run from channel messages.

CVSS assessment

Issue	CVSS v3.1	Vector
Remote `/bridge spawn` shell execution	8.8 High	`AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H`

Rationale:

- Attack complexity is low once the sender is accepted by a configured remote channel/gateway.
- Privileges are bounded to a remote gateway/channel sender rather than an unauthenticated internet user.
- The impact is high because the reachable sink is arbitrary shell command execution as the OpenHarness process user.

Safe reproduction steps

On a vulnerable build, with a remote channel/gateway sender accepted by configuration:

1. Send a bridge spawn command from the remote channel:

```
/bridge spawn id; pwd
```

2. Observe that the gateway accepts the command and reports a spawned bridge session.
3. Retrieve output through the bridge output path:

```
/bridge list
/bridge output <SESSION_ID>
```

A local validation harness can also prove the same boundary by checking that `/bridge` was registered as remote-invocable and that `/bridge spawn ...` starts a shell subprocess. I re-ran that harness against both the vulnerable base and this branch:

```
# origin/main at 380bab4
REMOTE_INVOCABLE True
REMOTE_ADMIN_OPT_IN False
FINAL Spawned bridge session bridge-... pid=...
BRIDGE_SESSIONS 1
MARKER_EXISTS True
MARKER_CONTENT REMOTE_BRIDGE_EXEC

# this PR branch
REMOTE_INVOCABLE False
REMOTE_ADMIN_OPT_IN True
FINAL /bridge is only available in the local OpenHarness UI.
BRIDGE_SESSIONS 0
MARKER_EXISTS False
```

Expected vulnerable behavior

- `/bridge` is considered remotely invocable.
- `/bridge spawn ...` reaches the command handler from the remote gateway path.
- The bridge manager spawns a process for the attacker-controlled command.

Changes in this PR

- Register `/bridge` with `remote_invocable=False`.
- Mark `/bridge` with `remote_admin_opt_in=True` so trusted deployments can still opt in explicitly using the existing gateway admin-command mechanism.
- Add registry tests asserting `/bridge` is local-only by default and eligible for explicit remote admin opt-in.
- Add a gateway regression test proving remote `/bridge spawn ...` is denied before the handler runs.

Files changed

Category	Files	What changed
Security boundary	`src/openharness/commands/registry.py`	Marks `/bridge` local-only by default and explicit-admin-opt-in capable
Tests	`tests/test_commands/test_registry.py`	Adds command metadata regressions for `/bridge`
Tests	`tests/test_ohmo/test_gateway.py`	Adds remote gateway denial regression for `/bridge spawn`

Maintainer impact

- Local UI usage of `/bridge` is preserved.
- Existing remote admin opt-in semantics are reused instead of adding a new configuration model.
- Deployments that do nothing get the secure default: bridge process spawning is not reachable from remote channel messages.
- Trusted operators can still explicitly opt into `/bridge` if they intentionally want remote bridge administration.

Fix rationale

The gateway already has a command-level remote execution boundary. The least surprising and lowest-risk fix is to apply that existing boundary to `/bridge`, because `/bridge spawn` is a local control-plane operation with direct process-execution effects.

Type of change

- [x] Security fix
- [x] Tests
- [ ] Documentation update
- [ ] Refactor with no behavior change

Test plan

Executed locally:

- [x] `PYTHONPATH=src:. uv run pytest -o addopts='' tests/test_commands/test_registry.py::test_bridge_command_is_marked_local_only tests/test_commands/test_registry.py::test_bridge_command_supports_explicit_remote_admin_opt_in`

`tests/test_ohmo/test_gateway.py::test_runtime_pool_blocks_bridge_spawn_from_remote_messages tests/test_ohmo/test_gateway.py::test_runtime_pool_blocks_local_only_commands_from_remote_messages tests/test_ohmo/test_gateway.py::test_runtime_pool_allows_opted_in_remote_admin_commands -q`

- [x] `PYTHONPATH=src:. uv run pytest -o addopts='' tests/test_commands/test_registry.py tests/test_ohmo/test_gateway.py -q`
- [x] `PYTHONPATH=src:. uv run python -m compileall -q src/openharness/commands/registry.py tests/test_commands/test_registry.py tests/test_ohmo/test_gateway.py`
- [x] `git diff --check`
- [x] `uv run ruff check src/openharness/commands/registry.py tests/test_commands/test_registry.py tests/test_ohmo/test_gateway.py`
- [x] Added-line secret scan: no findings
- [x] External reproducibility harness against `origin/main` at `380bab4`: marker file was written with `REMOTE_BRIDGE_EXEC`
- [x] External reproducibility harness against this PR branch: `/bridge` denied, no bridge sessions created, marker file absent

Note: `uv run ruff format --check` reports pre-existing whole-file formatting churn for the touched legacy files, so this PR keeps the diff minimal and does not include unrelated formatting-only rewrites.

Disclosure notes

- This PR is intentionally bounded to a gateway-adjacent variant of the already-public remote slash-command trust-boundary family in **fix: harden gateway slash command security** #127.
- The attacker model is an accepted remote channel/gateway sender, not an arbitrary unauthenticated internet user.
- No production system was tested.
- The patch addresses the command metadata boundary and adds regression tests for the remote gateway denial path.

`fix: keep bridge command local-only by default` ✓ `438e373`



tjb-tech merged commit **438e373** into `HKUDS:main` 4 days ago

4 checks passed

View details



Hinotoi-agent mentioned this pull request 4 days ago

[security] test(gateway): cover bridge spawn repro path #209

Merged

11 tasks

Reviewers

No reviews

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

2 participants