[security] fix(sessions): validate session_id before deleting session files #409

Closed **Hinotoi-agent** wants to merge 1 commit into `nesquena:master` from `Hinotoi-agent:fix/session-delete-…`

Hinotoi-agent commented last week • edited

Contributor

Summary

This PR fixes an arbitrary file deletion issue in the WebUI session delete API.

The current `/api/session/delete` handler trusts the client-supplied `session_id` when building the on-disk session file path. Because that value is used directly in:

```
SESSION_DIR / f"{sid}.json"
```

an attacker can supply an absolute path or traversal-style value and cause the server to unlink files outside the session directory.

This patch validates session identifiers before they are used in path construction, resolves the candidate delete target, enforces containment under `SESSION_DIR`, and adds regression coverage for absolute-path and traversal payloads.

Security issues covered

Issue	Impact	Severity
Arbitrary file deletion via unvalidated `session_id` in `/api/session/delete`	Filesystem boundary break allowing deletion of writable `.json` files outside the WebUI session store	High

Before this PR

- `/api/session/delete` accepted arbitrary `session_id` values from the request body.
- The delete target was built directly from untrusted input.
- Absolute-path payloads could bypass `SESSION_DIR` entirely.
- Traversal-style payloads were not rejected before filesystem use.
- The route could unlink writable `.json` files outside the intended session store.

After this PR

- `session_id` values are validated before path construction.
- Invalid absolute or traversal-style payloads are rejected with a client error.
- The candidate delete target is resolved and checked for containment under `SESSION_DIR`.
- The route can only delete legitimate session files inside the session store.
- Regression tests pin both absolute-path and traversal rejection behavior.

Why this matters

The session delete API is intended to delete only WebUI session files.

In its vulnerable form, it can be turned into a general file deletion primitive for attacker-controlled `.json` targets writable by the server process. That breaks the trust boundary around the WebUI session store and can affect WebUI state, Hermes state, or other writable JSON-backed files on the host.

If the WebUI is exposed without authentication, this becomes remotely reachable. In authenticated deployments, it remains a strong authenticated arbitrary-file-delete vulnerability.

Attack flow

```
attacker sends POST /api/session/delete
    -> supplies crafted session_id such as /tmp/victim-secret
        -> server builds SESSION_DIR / f"{sid}.json"
            -> pathlib discards SESSION_DIR when sid is absolute
                -> unlink() runs on attacker-controlled target outside SESSION_DIR
```

Affected code

Area	Files
Session deletion route	`api/routes.py`

Area	Files
Regression coverage	`tests/test_sprint3.py`

Root cause

- `session_id` supplied by the client was treated as a safe filesystem path component.
- The route built `SESSION_DIR / f"{sid}.json"` directly from untrusted input.
- The delete path did not reuse the session-id validation already present in `Session.load()`.
- No resolved-path containment check was enforced before calling `unlink()`.

CVSS assessment

Issue	CVSS v3.1	Vector
Arbitrary file deletion via unvalidated `session_id` in `/api/session/delete`	8.1 High	`CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:H`

Rationale:

- A low-privileged authenticated caller can trigger the vulnerable route directly.
- No victim interaction is required.
- The bug provides a filesystem deletion primitive outside the intended session-store boundary.
- Integrity and availability impact are both meaningful because writable host-side JSON files can be removed.

Safe reproduction steps

1. Start the WebUI with an isolated state directory.
2. Create a writable file outside `SESSION_DIR`, for example:
   `/tmp/victim-secret.json`
3. Send:
   `POST /api/session/delete`
   with:
   `{"session_id":"/tmp/victim-secret"}`
4. Observe the vulnerable behavior:
   - the API returns `{"ok": true}`
   - `/tmp/victim-secret.json` is deleted

Expected vulnerable behavior

- A file outside `SESSION_DIR` is deleted even though the endpoint is only supposed to delete session files.
- The route reports success because it treats the attacker-controlled target as if it were a normal session file.

Changes in this PR

- validate `session_id` before path construction
- reject unsafe session identifiers early
- resolve the candidate delete target before unlinking
- enforce containment under `SESSION_DIR`
- preserve session index invalidation behavior
- add regression tests for absolute-path and traversal payloads

Files changed

Category	Files	What changed
Route hardening	`api/routes.py`	Added session-id validation and resolved-path containment enforcement before delete
Regression tests	`tests/test_sprint3.py`	Added absolute-path and traversal payload coverage for `/api/session/delete`

Maintainer impact

- The patch is narrow and limited to session-delete input validation and path containment.
- It does not change session storage format, chat execution, streaming, auth behavior, or profile behavior.
- Existing valid session deletion behavior remains intact.

Fix rationale

The safest fix is to align this route with the project's existing session-id trust model instead of treating delete as a special case.

Validating the identifier and enforcing resolved-path containment ensures the route can only operate on legitimate session files inside the intended session-store root.

Type of change

- [x] Security fix

- [x] Bug fix
- [x] Tests
- [ ] New feature
- [ ] Breaking change
- [ ] Documentation-only change

Test plan

- [x] `python -m pytest tests/test_sprint3.py -q -k 'session_delete'`
- [x] `python -m pytest tests/test_sprint1.py -q -k 'test_session_delete'`
- [x] `python -m pytest tests/test_regressions.py -q -k 'delete'`

Added regression coverage for:

- absolute-path payload rejected in `/api/session/delete`
- traversal-style payload rejected in `/api/session/delete`
- valid session deletion still succeeds

Disclosure notes

- Claims in this PR are limited to the reviewed `/api/session/delete` code path and reproduced file-deletion behavior.
- This PR does not claim broader arbitrary file read/write impact across other routes.
- The fix is intentionally limited to the delete path trust boundary so the change remains easy to review and hard to regress.

 `fix(sessions): validate session_id before deleting files` ✓ `34495a7`

 **nesquena-hermes** mentioned this pull request last week

[security] fix(sessions): validate session_id before deleting session files (#409) #412

Merged

  **nesquena-hermes** closed this in #412 last week

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

1 participant