MCP HARDENING RECORD

Federal Register MCP Testing Record

Federal Register MCP hardened across four audit rounds including a retroactive deep audit

Skill
federal-register-mcp v0.2.2
github.com/1102tools/federal-register-mcp

Date
April 2026
1102tools.com

Executive Summary

This Model Context Protocol server exposes the Federal Register API as 8 callable tools for rulemaking tracking, FAR case history, comment-period monitoring, and regulatory research since 1994. It was hardened across four audit rounds including an initial hardening pass that fixed 17 findings and a retroactive deep audit that surfaced 12 more hidden bugs. The signature finding was a pydantic validation crash on every `list_agencies` call that was invisible to the prior unit tests because they awaited raw coroutines instead of invoking through the FastMCP pipeline. The MCP ships with 77 regression tests (64 offline plus 13 live-gated).

Metric	Value
MCP tools exposed	8
Total regression tests	77 (64 offline, 13 live-gated)
Audit rounds completed	4
Total items addressed	30 (12 P1 plus 3 P2 plus 2 P3 initial, plus 12 more from deep audit)
Current release	0.2.2
PyPI status	Published as `federal-register-mcp`, auto-publishes via Trusted Publisher on tag push

What Was Tested

The MCP exposes 8 tools covering the Federal Register API surface. Testing covered all of them end-to-end.

Core: `search_documents`, `get_document`, `get_documents_batch`, `get_facet_counts`

Reference: `list_agencies`, `get_public_inspection`, `open_comment_periods`

Workflow: `far_case_history`

Each tool was exercised for argument validation, input sanitization, date format checking, empty-input detection (empty strings, empty arrays, whitespace-only), payload size bounds, response-shape

guarantees, error translation (including stripping HTML from 404 bodies), and real-world data handling against the live public Federal Register API.

How It Was Tested

Testing discipline

Prior unit tests in v0.1.x awaited raw coroutines directly, which bypassed the FastMCP tool pipeline and its pydantic return validation. This was the failure mode that hid the `list_agencies` crash: FastMCP validates every tool's return against its type annotation, but the 0.1.1 smoke test awaited the raw coroutine and never triggered the validation layer. The hardening program switched to invoking tools through `mcp.call_tool(name, kwargs)` the way a real MCP client does. That change surfaced the pydantic crash that had been present since v0.1.

Audit rounds

Release	Context	Findings
0.1.1	Initial release with unit tests	Smoke test said OK (wrong)
0.2.0	First real hardening pass through `mcp.call_tool` pipeline	17 findings fixed (list_agencies crash, payload bombs, basic validation)
0.2.1	Cross-MCP `extra='forbid'` back-port from sam-gov-mcp 0.3.1	1 cross-fix applied
0.2.2	Retroactive deep audit with live-gated suite	12 additional findings including `far_case_history` substring match, `get_documents_batch` empty arrays, `get_facet_counts` validation gaps

Live audit status

The repository includes 13 live-gated regression tests executable via `FR_LIVE_TESTS=1 pytest` covering real document search, single-document fetch, batch fetch, facet counts, agency listing, public inspection queue, open comment periods, and FAR case history. Federal Register is a free, public API with no authentication requirement.

Issues Found and Fixed

Priority 1: Crashes and silent wrong data

Fourteen items in this class across initial hardening and deep audit.

Crashes:

Issue	Fix
`list_agencies` pydantic crash on every call. The return type was declared `dict[str, Any]` but the API returns a bare list of 470 agencies. FastMCP validates returns against pydantic schemas and threw a `DictModel` validation error on every call. Prior unit tests missed this because they awaited the raw coroutine and bypassed the validation layer.	Return type changed to `list[dict[str, Any]]` with optional slim-mode field set. Regression test invokes via `mcp.call_tool` to exercise the full pipeline.

Payload bombs (known at shipping, fixed in 0.2.0):

Issue	Fix
`get_public_inspection` returned a 170KB payload on unfiltered calls.	Slim-mode field set by default; `include_detail=True` opt-in for full payload.
`open_comment_periods` returned 188KB with a hardcoded `per_page=100`.	`per_page` bounded and slim-mode applied.
`list_agencies` returned a 700KB dump of all 470 agencies with full descriptions.	Slim fields by default plus optional `query` filter and `include_detail` flag.
`search_documents` allowed `per_page=1000`. A single response of roughly 2MB could blow the MCP token cap.	Lowered to `per_page <= 100` with documented guidance.

Silent wrong data from deep audit (0.2.2):

Issue	Fix
`far_case_history(docket_id="x")` returned 65 unrelated documents via 1-character substring match across all dockets.	Minimum 3-character docket_id enforced; substring match replaced with prefix match for short inputs.
`get_documents_batch` with empty entries `["", "", ""]` sent `,,` to the API which returned all 10,000 documents.	Empty and whitespace-only entries rejected; at-least-one-non-empty enforced.
`search_documents` with a 10,000-character term triggered an HTTP 414 URI Too Large and the raw HTML response body leaked to the caller.	Term length capped at 500 characters; `_clean_error_body` helper strips HTML from error responses.
`get_facet_counts` accepted any string as a date. `"2026/01/01"` was silently ignored (filter dropped) and the tool returned unfiltered data.	Date format enforced as `YYYY-MM-DD` with actionable error.
`get_facet_counts agencies=[]` silently ignored the filter and returned the full agency aggregate.	Empty-list rejected; at-least-one-filter enforced.
`get_facet_counts` reversed date range (`gte > lte`) silently returned `{}` instead of an error.	Reversed ranges raise actionable error.
`get_facet_counts` with no filters returned the entire all-time aggregate dataset.	At-least-one-filter guard enforced.
`search_documents docket_id=" "` (whitespace-only) was treated as the "All Documents" filter.	Whitespace stripped and empty-after-strip rejected.
`search_documents term=" "` had the same problem and returned everything.	Same whitespace handling applied.

Priority 2: Validation gaps

Ten items in this class:

Issue	Fix
`search_documents` reversed date ranges silently returned 0 documents.	Reversed ranges raise actionable error.
`search_documents` bad date formats silently passed through (YYYY-MM-DD validator missing).	Validator added.
`search_documents agencies=[]` and `doc_types=[]` silently ignored.	Empty-list rejected.
`search_documents per_page` and `page` were unbounded.	Clamped.
404 responses leaked raw `<html xmlns...>` body to the caller.	`_clean_error_body` helper strips HTML.
`search_documents docket_id` and `regulation_id_number` accepted arbitrarily long strings (future 414 risk).	Length-clamped.
`get_facet_counts` had parallel validation gaps as `search_documents`.	Same validators applied.
`list_agencies query=""` returned all 470 agencies (fine), but `query=" "` returned 0 (inconsistent).	Whitespace stripped; empty-after-strip returns all.
`get_public_inspection keyword_filter=""` matched everything.	Empty-after-strip treated as "no filter" with clear output flag.
`get_document` accepted `#`, `?`, `&`, `%` in document numbers and produced confusing 404s.	`_validate_doc_number` helper enforces valid shape.

Priority 3: Cleanup items

Six items including a stale `USER_AGENT = "federal-register-mcp/0.1.1"`, misleading hard-coded "Verify the document_number" text in 404 messages for agency and facet endpoints, pre-1994 dates silently accepted (Federal Register did not exist before 1994), inconsistent `_strip_or_none` vs length-clamp vs WAF-style validation across tools, and the lack of a live-gated regression suite before 0.2.2. All resolved.

Helpers added (0.2.2)

The 0.2.2 deep audit introduced a consistent set of input and response helpers reused across all 8 tools: `_strip_or_none`, `_require_min_length`, `_clamp_str_len`, `_validate_doc_number`, `_warn_pre_fr_date`, `_clean_error_body`. Extended date validation, empty-list rejection, and page/per_page clamps were propagated into `get_facet_counts`. The `per_page` cap was lowered from 1000 to 100.

Test Coverage

The repo ships 77 regression tests. All 77 pass on every release cycle.

File	Purpose	Test count
`tests/test_validation.py`	Main regression suite covering every round finding, plus 13 live-gated integration tests	77
`tests/stress_test.py`	Round 1 scenarios (retained for reproducibility)	N/A (scenario script)
`tests/stress_test_deep.py`	0.2.2 deep-audit scenarios including `far_case_history` substring match, `get_documents_batch` empty arrays, `get_facet_counts` validation (retained for reproducibility)	N/A (scenario script)

Regression tests invoke tools through the FastMCP registry (`mcp.call_tool`). This is the discipline that surfaced the `list_agencies` pydantic crash. An autouse fixture resets the shared httpx client between tests.

Release History

Version	Focus	Outcome
0.1.1	Initial release with unit tests awaiting raw coroutines; smoke test reported OK but missed the `list_agencies` crash	Baseline (with latent bug)
0.2.0	First real hardening pass through `mcp.call_tool` pipeline: 17 findings fixed including the pydantic crash and payload bombs	List-agencies crash fixed; payload-bomb slim-mode pattern introduced
0.2.1	Cross-MCP `extra='forbid'` back-port from sam-gov-mcp 0.3.1	+1 regression test
0.2.2	Retroactive deep audit with live-gated suite: 12 additional findings fixed; 77 regression tests including 13 live-gated	`far_case_history` substring, empty-array batch, facet-count validation gaps resolved

Cross-MCP Context

This MCP is one of eight servers in the 1102tools federal-contracting MCP suite (`bls-oews-mcp`, `ecfr-mcp`, `gsa-calc-mcp`, `gsa-perdiem-mcp`, `regulationsgov-mcp`, `sam-gov-mcp`, `us-aspending-gov-mcp`, and this one). All eight were hardened under the same playbook. Patterns reused or confirmed here:

- **"Raw coroutine vs MCP pipeline" testing discipline** was most clearly demonstrated here. The `list_agencies` pydantic crash was present on every call since v0.1 but was invisible to tests that awaited the raw coroutine. Only invoking through `mcp.call_tool` triggered the FastMCP return validation.
- **Payload-bomb slim-mode pattern** (`include_detail=False` by default, opt-in for full payload) was applied here to `list_agencies`, `get_public_inspection`, and `open_comment_periods`, and reused across the suite.
- **`extra='forbid'` cross-fix** back-ported from sam-gov-mcp 0.3.1 in 0.2.1.
- **`_clean_error_body` helper** to strip HTML from 404 responses was codified here and reused in other MCPs that face similar raw-HTML leaks.

What Was Not Tested

- **Rate-limit behavior.** Federal Register does not document strict rate limits. The MCP surfaces any upstream limits but does not implement client-side throttling.
- **Pre-1994 documents.** Federal Register did not exist before 1994. The MCP now warns on pre-1994 dates but does not predict which specific pre-1994 queries will return zero.
- **Historical agency reorganizations.** When an agency is merged or renamed, historical documents may appear under the old agency slug. The MCP passes through the Federal Register API's agency mapping but does not back-fill historical relationships.
- **Public inspection queue timing.** The public-inspection queue changes throughout the day. The MCP returns the queue as of the call; real-time updates are not pushed.

Verification

All testing artifacts are in the repository. The methodology and fixes are reviewable commit-by-commit in git history. The regression test suite runs via `pytest` in the repo root and can be re-executed by anyone. The live suite runs with `FR_LIVE_TESTS=1 pytest` and requires no API key (Federal Register is a free, public API).

> **Testing Methodology**
>
> Evaluators: James Jenrette, 1102tools, with Claude Code Opus 4.7 (1M context, max effort, Claude Max 20x subscription) during the hardening playbook execution.
>
> Testing spanned four rounds from initial hardening (17 findings including the pydantic crash and payload bombs) through cross-MCP `extra='forbid'` application to a retroactive deep audit (12 additional findings). The live regression suite runs against the production Federal Register API when enabled with `FR_LIVE_TESTS=1`.
>
> Test count: 77 regression tests. Total items addressed: roughly 30. Current version: 0.2.2. PyPI: `federal-register-mcp`.
>
> Source: github.com/1102tools/federal-register-mcp. License: MIT.