MCP HARDENING RECORD

# eCFR MCP Testing Record

*Regulatory text MCP hardened across five audit rounds against the live eCFR API*

**Skill**
ecfr-mcp v0.2.1
github.com/1102tools/ecfr-mcp

**Date**
April 2026
1102tools.com

## Executive Summary

This Model Context Protocol server exposes the eCFR (Electronic Code of Federal Regulations) API as 13 callable tools covering regulatory text, structure, search, version history, and common acquisition workflows. It was hardened across five audit rounds that surfaced and fixed 72 bugs, including two catastrophic silent wrong-data paths and multiple 23MB payload bombs triggered by empty-string inputs. The MCP ships with 101 regression tests (88 offline plus 13 live-gated) that run on every change and can be executed against the real public eCFR API on demand.

| Metric | Value |
|---|---|
| MCP tools exposed | 13 |
| Total regression tests | 101 (88 offline, 13 live-gated) |
| Audit rounds completed | 5 |
| P0 catastrophic bugs found and fixed | 2 |
| P1 silent-wrong-data bugs found and fixed | 26 |
| P2 validation gaps found and fixed | 32 |
| P3 cleanup items found and fixed | 12 |
| Current release | 0.2.1 |
| PyPI status | Published as `ecfr-mcp`, auto-publishes via Trusted Publisher on tag push |

## What Was Tested

The MCP exposes 13 tools covering the eCFR API surface. Testing covered all of them end-to-end.

**Core endpoints:** `get_latest_date`, `get_cfr_content`, `get_cfr_structure`, `get_version_history`, `get_ancestry`, `search_cfr`, `list_agencies`, `get_corrections`

**Workflow convenience:** `lookup_far_clause`, `compare_versions`, `list_sections_in_part`, `find_far_definition`, `find_recent_changes`

Each tool was exercised for argument validation, input sanitization, response-shape guarantees, error translation, URL construction, XML parser robustness, and real-world data handling against the live public eCFR 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 validation layer. This skipped whole categories of bugs. The hardening program switched to invoking tools through `mcp.call_tool(name, kwargs)` the way a real MCP client does. That change alone surfaced the `search_cfr Po` where every filter argument was silently dropped.

### Audit rounds

| Round | Scope | Probe count | Finding class |
|---|---|---|---|
| 1 | Live probing across all 13 endpoints | 43 probes | 2 P0, 17 P1, 18 P2, 6 P3 |
| 2 | Response-shape fragility and mock fuzzing | 15 probes | 15 new shape-fragility bugs |
| 2b | XML parser pathological inputs | 7 probes | 7 parser bugs (non-string input, CDATA, HTML comments, case sensitivity, attribute handling) |
| 3 | URL construction, injection vectors, concurrency | 5 probes | 5 new bugs |
| 4 | Static review, concurrency stability | 5 probes | 5 new cleanup items |
| 5 | JSON decode paths, non-JSON response handling | 7 probes | 7 additional HTTP edge cases |

### Live audit status

All rounds included live calls against the production eCFR API. The repository includes 13 live-gated regression tests executable via `ECFR_LIVE_TESTS=1 pytest` covering real search, content retrieval, FAR clause lookup, structure navigation, version history, and corrections. No API key is required; eCFR is a free, public API.

## Issues Found and Fixed

### Priority 0: Catastrophic silent wrong-data bugs

Two bugs in this class, both in the search pipeline. Every caller of `search_cfr` or `find_recent_changes` received random default results that appeared legitimate.

| Issue | Fix |
| --- | --- |
| `search_cfr` silently dropped every filter argument. The code built the query string into the URL path via `_get_json(f"...?{qs}")` then passed `params={}` to httpx. httpx strips any existing query string when `params` is provided, even if empty. Every value for `query`, `title`, `chapter`, `part`, `subpart`, `section`, `current_only`, `last_modified_after`, `last_modified_before`, `per_page`, and `page` was silently ignored. The API returned a random default 20-result response every call. Verified: four different query strings all returned the same shape and count. | Helper no longer passes `params={}`; query string is constructed once and passed intact. Regression test calls `search_cfr` with four distinct query values and asserts different result sets. |
| `find_recent_changes` delegated to `search_cfr` and inherited the Po. `since_date` was silently dropped along with every other filter. | Fixed by the `search_cfr` repair plus an explicit regression test for `find_recent_changes` with dated filters. |

## Priority 1: Silent wrong-data bugs

Twenty-six bugs in this class, grouped below by theme.

### Payload bombs on empty-string inputs (4 bugs):

| Issue | Fix |
|---|---|
| `get_cfr_content(section="")` returned the entire 23.2MB Title 48 XML document. Same for `section=" "` after internal strip. | Empty-string and whitespace-only inputs rejected up front with a clear error pointing to valid section-id shapes. |
| `lookup_far_clause(section_id="")` forwarded to `get_cfr_content` and produced the same 23.2MB payload bomb. | Rejected locally before the HTTP call. |
| `find_far_definition(term="")` matched every paragraph in FAR 2.101 and returned 437KB. `term="the"` returned 327KB from 358 matches. | Minimum 2-character term enforced; regression tests cover both zero-length and single-character cases. |
| `get_cfr_content(chapter="0")`, `chapter="27"`, `chapter=""` silently returned the full Title 48 XML. No validation against the `TITLE_48_CHAPTERS` constant. | Chapter validated against the known chapter set before the HTTP call. |

### Oversized known payloads (3 bugs):

| Issue | Fix |
|---|---|
| `list_agencies` returned 98KB of JSON including 153 agencies with deep `cfr_references` trees. | Documented response size and added a `flat=True` convenience option that returns only top-level agency summary. |
| `get_corrections` for Title 48 returned 109KB (283 corrections) without pagination. | Added optional date-range and per-part filters; documented expected size. |
| `compare_versions` for large sections returned 100 to 150KB × 2 per call. | Documented; caller must size accordingly. No size ceiling imposed since comparison requires both payloads. |

### Type rejection on common LLM patterns (3 bugs):

| Issue | Fix |
|---|---|
| Pydantic schema rejected `part=15` as integer. LLMs commonly pass integer chapter and part numbers. | All integer-convertible parameters coerce `int` to `str` before forwarding. Covers `get_ancestry`, `get_cfr_structure`, and `list_sections_in_part`. |

### Reserved title null handling (2 bugs):

| Issue | Fix |
|---|---|
| Title 35 is reserved; the API returns `up_to_date_as_of: None`. `_resolve_date` returned None silently and the URL became `/full/None/title-35.xml`, producing a cryptic 404. | Reserved titles now raise a clear "Title N is reserved and has no regulatory content" error with pointers to valid titles. |
| `get_latest_date` for a reserved title returned `{"up_to_date_as_of": null}` with no warning. | Explicit null check added; reserved titles now return a reason field. |

### Raw dict access fragility (3 bugs):

| Issue | Fix |
|---|---|
| `title["number"]` raw access would crash with `KeyError` if the API schema shifted. | `.get()` with sensible defaults throughout. |
| `node.get("children", []) or []` was missing. If the API returned `"children": null`, the enclosing code crashed with `TypeError`. | Explicit null-coalescing on every children access. |
| `_parse_xml_to_text` did not HTML-unescape headings. `&`, `<`, numeric entities like `&#38;` leaked through verbatim. Paragraph bodies were unescaped but headings were not. Citation attributes were also not unescaped. | Unified HTML unescape across all parsed text nodes, including headings, citations, and paragraph bodies. |

### HTTP error handling (3 bugs):

| Issue | Fix |
|---|---|
| Empty or malformed 200 bodies caused `r.json()` to raise `JSONDecodeError`, leaking as unfriendly traceback. | `_decode_json_response` helper catches and re-raises with API context. |
| 404 HTML pages served as `text/html` broke `r.json()` the same way. | Content-type inspection before decode; HTML bodies produce a clear "eCFR returned HTML at {path}, expected JSON" error. |
| `None` response passed through and downstream `.get()` crashed with `AttributeError`. | `_ensure_dict_response` helper now guarantees a dict return from every helper. |

### Response-shape fragility from round 2 (8 bugs):

| Issue | Fix |
|---|---|
| `_resolve_date` crashed if the returned `data` was a list, string, int, or None rather than a dict. | Type-checked before member access. |
| List entries that were None or bare strings crashed at `title["number"]`. | Defensive iteration filters non-dict entries before access. |
| If the API returned `title["number"]` as int instead of string, equality check against a string argument silently failed. | Numeric and string forms normalized before comparison. |
| `_walk_structure(node)` crashed when the node was not a dict, or when children contained dict or None entries. | Guard clauses added throughout the recursive walker. |
| `list_sections_in_part` crashed if the API returned None. | Return-shape guard added. |
| Pass-through endpoints (`search_cfr`, `get_version_history`, `list_agencies`, `get_corrections`, `get_ancestry`, `get_cfr_structure`) returned whatever the API gave without type annotation enforcement. | All pass-throughs now validate shape and wrap with `_ensure_dict_response` or `_ensure_list_response`. |

### XML parser pathological inputs from round 2b (7 bugs):

| Issue | Fix |
|---|---|
| Non-string input (bytes, None, int) to `_parse_xml_to_text` raised `TypeError`. | Type check at entry; non-string input returns empty string with a logged warning. |
| CDATA sections were handled as tags: content was preserved but `]]>` artifacts leaked into output. | CDATA blocks now unwrapped cleanly. |
| HTML comment blocks `<!-- ... -->` were stripped as tags, so comment bodies leaked into the output text. | Comments recognized and removed entirely. |
| Mixed-case `<HEAD>`, `<P>`, `<head>`, `<p>` tags were not recognized (regex was case-sensitive). | Parser matching made case-insensitive. |
| Attribute-bearing heads like `<HEAD class="..."` were dropped because the regex required no attributes. | Attribute-tolerant matching added. |
| Citation attributes were dropped entirely. | Citation attributes now captured and emitted alongside the citation text. |
| Numeric entities `&#38;` in heading text were not unescaped. | Unified unescape covers numeric, named, and hex entities. |

## Priority 2: Validation gaps

Thirty-two bugs in this class. Representative items:

| Issue | Fix |
|---|---|
| `get_cfr_content` accepted `date=""`, `date=" "`, `date="2026/04/16"`, `date="April 16, 2026"`, `date="current"`, `date="2026--04-16"`. All forwarded as cryptic 404s. | `YYYY-MM-DD` regex enforced at the arg layer. |
| `compare_versions` accepted the same malformed date shapes for `date_before` and `date_after`. | Same regex applied to both. |
| `find_recent_changes` since_date accepted malformed shapes with the same result. | Same regex applied. |
| `search_cfr` accepted `per_page=0`, `per_page=-1`, `page=0`, `page<0`. | Bounded `per_page` to 1..200 and `page` to >= 1 with actionable errors. |
| `search_cfr` accepted `query=""` which, once the P0 was fixed, became a massive-payload problem. | Minimum 2-character query enforced. |
| `search_cfr` query length was unbounded; 2000+ character strings were forwarded. | Capped at 500 characters with a pointer toward narrowing filters. |
| `get_cfr_content` section, part, subpart, chapter arguments were not stripped and not empty-checked. `section=" "` hit the API with `%20%20%20`. | `.strip()` and empty check applied to all identifier args. |
| `find_far_definition(term="")` was not rejected. Single-character terms returned junk. | Minimum 2-character term enforced. |
| `get_latest_date` accepted any integer title number, including negative and zero. | Bounded 1 to 50. |
| `compare_versions` accepted `date_before == date_after` (trivial no-op). | Reversed and equal date ranges raise actionable errors. |
| Common LLM shapes like `section="FAR 15.305"`, `section="48 CFR 15.305"`, `section="15.305(a)"`, `section=" 15.305 "` all hit the API as 404s. | Prefix normalization strips "FAR ", "48 CFR ", subparagraph suffixes, and whitespace before forwarding. |

| Issue | Fix |
|---|---|
| `TITLE_48_CHAPTERS`, `COMMON_FAR_SECTIONS`, and `SEARCH_MAX_TOTAL` constants were defined but unused. | All three now wired into their respective validators. |
| `_format_error` was not robust to non-string body; `body.lower()` crashed on bytes. | Type check before normalization. |

### Priority 3: Cleanup items

Twelve items in this class, including a stale USER_AGENT header at `ecfr-mcp/0.1.1`, an `import json` inside a function body, a regex that matched only the first `<HEAD>` in sections with multiple heads, nested `<I><E>` producing mismatched markdown, and missing timeout consistency across helpers. All resolved.

### Response-shape defense

The `_get_json` helper now guarantees a dict or list return via `_ensure_dict_response` and `_ensure_list_response`. eCFR normally returns well-formed JSON for the endpoints this MCP uses; anything else (None, bare string, HTML error page from a proxy) previously leaked as an unhelpful type-confusion crash. It now surfaces clearly as "eCFR returned empty body at {path}" or "unexpected {type} at {path}".

## Test Coverage

The repo ships 101 regression tests across the test folder. All 101 pass on every release cycle.

| File | Purpose | Test count |
|---|---|---|
| `tests/test_validation.py` | Main regression suite covering every round-1 through round-5 finding, plus 13 live-gated integration tests | 101 |
| `tests/stress_test.py` | Round 1 live-probe scenarios (retained for reproducibility) | N/A (scenario script) |
| `tests/stress_test_r2.py` | Round 2 response-shape fuzzing (retained for reproducibility) | N/A (scenario script) |
| `tests/stress_test_xml.py` | Round 2b XML parser pathological inputs (retained for reproducibility) | N/A (scenario script) |

Regression tests invoke tools through the FastMCP registry (`mcp.call_tool`) rather than awaiting decorated coroutines directly. This catches bugs in the tool pipeline that raw-coroutine tests miss. An autouse fixture resets `srv._client` between tests so the shared httpx client does not leak across event loops, preventing flaky results from async state carryover.

## Release History

| Version | Focus | Outcome |
|---|---|---|
| 0.1.1 | Initial release: 13 tools with basic unit tests | Basic coverage |
| 0.2.0 | Full 72-bug fix across 5 audit rounds using `mcp.call_tool()` pipeline; added 101 regression tests including 13 live-gated | 2 P0, 26 P1, 32 P2, 12 P3 resolved |
| 0.2.1 | Cross-MCP fix: pydantic `extra='forbid'` applied to every tool arg model to prevent typo'd-parameter silent filter-drop bugs (back-ported from sam-gov-mcp 0.3.1) | +1 regression test |

## Cross-MCP Context

This MCP is one of eight servers in the 1102tools federal-contracting MCP suite (`bls-oews-mcp`, `federal-register-mcp`, `gsa-calc-mcp`, `gsa-perdiem-mcp`, `regulationsgov-mcp`, `sam-gov-mcp`, `usaspending-gov-mcp`, and this one). All eight were hardened under the same playbook. Patterns that originated or propagated through this MCP:

- **The "23MB payload bomb on empty string" pattern** was codified here and became the template for similar payload-bomb fixes in other MCPs where empty-string inputs silently matched wildcard shapes.
- **The reserved-title null-handling pattern** informed gsa-perdiem-mcp's handling of no-data API responses for remote OCONUS locations.
- **`extra='forbid'` on pydantic arg models** was discovered during the sam-gov-mcp 0.3.1 audit. Applied here in 0.2.1 and to every other MCP in the suite.

## What Was Not Tested

- **Rate-limit behavior.** eCFR does not document rate limits publicly. The MCP passes through whatever limits the API enforces but does not implement client-side throttling. Heavy concurrent use may hit limits the MCP cannot anticipate.
- **Historical API changes.** Tests validate behavior against the current eCFR API. Breaking changes upstream (field renames, endpoint deprecations) are not caught by offline tests. Live-gated tests will catch them but must be run manually.
- **Titles outside 1 through 50.** The API's reserved-title list may shift if Congress enacts legislation reactivating a reserved title. The MCP's reserved-title check lags API changes until refreshed.
- **Historical dates before 2017.** eCFR's daily snapshot coverage is thinner in older years. The MCP surfaces upstream 404s but does not predict which dates are missing.

## 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 `ECFR_LIVE_TESTS=1 pytest` and requires no API key (eCFR 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 five rounds from integration stress through response-shape fuzzing, XML parser pathological inputs, URL and injection probes, and concurrency stability. The live regression suite runs against the production eCFR API when enabled with `ECFR_LIVE_TESTS=1`.

Test count: 101 regression tests. P0 catastrophic bugs found and fixed: 2. P1 bugs found and fixed: 26. P2 validation gaps closed: 32. P3 cleanup items closed: 12. Total findings: 72. Current version: 0.2.1. PyPI: `ecfr-mcp`.

Source: github.com/1102tools/ecfr-mcp. License: MIT.