New issue

Server-Side Request Forgery Vulnerability in mcp-browser-agent #25

Open

fkrcarry opened 2 weeks ago

mcp-browser-agent Vulnerability Audit Report

1) CNA / Submission Type

- Submission type: Report a vulnerability (CVE ID request)
- Reporter role: Independent security researcher
- Report date: March 21, 2026

2) Reporter Contact (fill before submit)

- Reporter name: `feng carry`
- Reporter email: `3405402279@qq.com`
- Permission to share contact with vendor: `Yes`

3) Vendor / Product Identification

- Vendor: imprvhub
- Product: mcp-browser-agent
- Repository: https://github.com/imprvhub/mcp-browser-agent
- Reviewed local source path: `datasets_set/001/datasets_001/imprvhub_mcp-browser-agent`
- Affected component(s):
- `src/handlers.ts`
- `src/executor.ts`

4) Vulnerability Type

- CWE: CWE-918 (Server-Side Request Forgery)
- Short title: SSRF vulnerability in MCP browser navigation

5) Affected Versions

- Confirmed affected: 0.8.0
- Suspected affected range: revisions containing the same request-to-sink flows listed below
- Fixed version: Not available at time of report (March 21, 2026)

6) Vulnerability Description

An input-handling flaw in mcp-browser-agent (src/executor.ts) allows an attacker who can reach the exposed MCP/HTTP interface to supply crafted data that flows from a request-controlled source to a security-sensitive sink without sufficient validation. This can lead to Server-Side Request Forgery (SSRF), enabling attackers to force the server to send requests to arbitrary URLs, potentially accessing internal systems or exfiltrating sensitive data.

7) Technical Root Cause

1. Request-controlled URL → network/browser sinks (SSRF class)
   - **Source:** `src/handlers.ts:65-69` — `CallToolRequestSchema` handler passes `request.params.name` and `request.params.arguments` into `executeToolCall(...)`.
   - **Routing:** `src/executor.ts:163-246` — `executeToolCall` dispatches by tool name; for API tools it calls `initApiClient(args.url)` (`src/executor.ts:179-180`, implementation `138-141`), and for `browser_navigate` it calls `handleBrowserNavigate`.
   - **Sinks (representative, all use attacker-controlled `args.url` without host/SSRF policy):**
     - **Browser navigation:** `src/executor.ts:248-274` — `handleBrowserNavigate` → `await page.goto(args.url, { timeout: args.timeout || 30000, waitUntil: args.waitUntil || "load" });`
     - **HTTP client (Playwright `APIRequestContext`):** e.g. `src/executor.ts:512-540` (`handleApiGet` → `client.get(args.url, ...)`), and analogously `handleApiPost` / `handleApiPut` / `handleApiPatch`.

8) Attack Prerequisites

- Attacker can invoke the MCP transport (e.g. `tools/call`) so that tool arguments reach `executeToolCall`.
- No effective URL allowlist, blocklist, or SSRF guard is applied to `args.url` before the sinks above.
- The server process can reach attacker-chosen targets (including RFC1918/link-local/cloud metadata endpoints where routing allows).

9) Proof of Concept / Reproduction Guidance

This proof of concept illustrates SSRF-style abuse via `browser_navigate`, which passes `url` into `page.goto` (see `src/executor.ts` / `src/tools.ts`).

1. Reproduction request (browser navigation SSRF probe; replace the URL with one valid in the deployment under test)

```
{"jsonrpc":"2.0","id":1,"method":"tools/call","params":{"name":"browser_navigate","argu
```

2. Validation

- Submit the request to the MCP interface of the affected deployment.
- Observe that the server/browser stack performs the request/navigation to the supplied `url` (e.g. response status/body in tool output, or successful navigation), confirming lack of SSRF restrictions on `args.url`.

10) Security Impact

- **Confidentiality:** High — SSRF can reach internal HTTP services or metadata endpoints and return content to the caller via tool responses.
- **Integrity / Availability:** Depends on reachable services (e.g. unsafe internal APIs); often Medium to High in combined SSRF scenarios.
- **Scope:** Unchanged unless chained with other issues; primary finding here is network-layer abuse from trusted MCP input.

11) CVSS v3.1 Suggestion

- Suggested vector: deployment-dependent (requires maintainer confirmation of effective trust boundary).
- Suggested base score: pending maintainer validation.

12) Workarounds / Mitigations

- Enforce a strict URL policy (allowlist of schemes/hosts, or deny private/link-local/metadata ranges) before any `page.goto` / `APIRequestContext` use.
- Require authentication, authorization, and auditing on MCP endpoints; rate-limit sensitive tools.
- Prefer outbound proxying with organizational policy over raw egress from the agent host.

13) Recommended Fix

- Block or validate `args.url` (and related inputs) at the MCP boundary and inside `executeToolCall` before calling `initApiClient`, `page.goto`, or `client.*` methods.

- Add regression tests that forbidden hosts/schemes cannot reach these sinks.
- Publish a maintainer security advisory once a patch is released.

14) References

- Repository: https://github.com/imprvhub/mcp-browser-agent
- Reviewed source file: `src/handlers.ts`
- Reviewed source file: `src/executor.ts`
- CWE-918 (SSRF): https://cwe.mitre.org/data/definitions/918.html

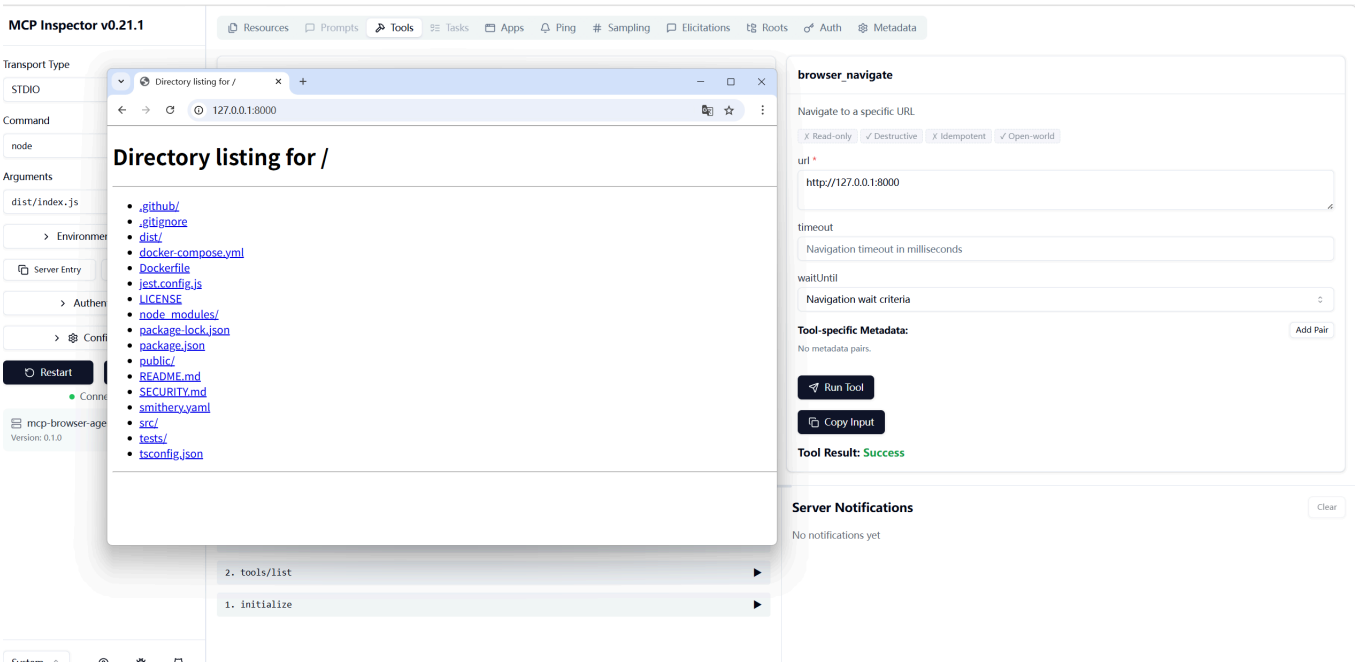
15) Credits

- Discoverer: `feng carry`
- Discovery method: Static analysis (CodeQL) plus repository source-code audit

16) Additional Notes for Form Mapping

- Audit verdict: Report sections 7–9 were revised to align with current `src/executor.ts` (tool parameters use `url`, not `command`; `api_*` tools use Playwright `APIRequestContext`, not shell execution).
- Dynamic exploit replay status: not completed in this batch run.
- Maintainer should validate release mapping before coordinated disclosure.

fkrcarry 2 weeks ago Author



Metadata

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with agent mode

No branches or pull requests

Participants