

BruceJqs / public_exp Public[Code](#) [Issues 34](#) [Pull requests](#) [Actions](#) [Projects](#) [Security and quality](#)[New issue](#)

Arbitrary File Write Vulnerability in hwpX-mcp-server of Dayoooun/hwpX-mcp #28

[Open](#)

BruceJqs opened 2 weeks ago

Owner

Arbitrary File Write Vulnerability in hwpX-mcp-server of Dayoooun/hwpX-mcp

1) CNA / Submission Type

- Submission type: Report a vulnerability (CVE ID request)
- Reporter role: Independent security researcher
- Report date: Apr 15, 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: Dayoooun
- Product: hwpX-mcp / hwpX-mcp-server
- Repository: <https://github.com/Dayoooun/hwpX-mcp>
- Affected component(s):
 - mcp-server/src/index.ts
 - src/mcp/server.ts

4) Vulnerability Type

- CWE: CWE-73 (External Control of File Name or Path)
- Short title: Arbitrary file write via MCP-controlled output path

5) Affected Versions

- Confirmed affected: 0.2.0
- Confirmed affected commit: `87850fd67f0488d79fcbf061a29938cae914a15d`
- Suspected affected range: revisions containing the same MCP request-to-filesystem flows listed below
- Fixed version: Not available at time of report

6) Vulnerability Description

An arbitrary file write vulnerability (CWE-73) has been identified in hwpix-mcp-server version 0.2.0 (commit 87850fd), specifically within the `save_document`, `export_to_text`, and `export_to_html` MCP tools. The server accepts caller-controlled `output_path` arguments and writes to those paths without validating that the destination resides inside a safe workspace, allowing parent-directory traversal or absolute paths. An attacker with network access to the MCP interface can create or overwrite files at arbitrary locations writable by the server process, leading to integrity loss, configuration corruption, or denial of service. No fixed version is available at the time of reporting.

7) Technical Root Cause

1. `js/file-access-from-request`

- Source: `mcp-server/src/index.ts:2156` (`CallToolRequestSchema` handler receives MCP tool arguments)
- Tool entry: `mcp-server/src/index.ts:4305` (`create_document` creates an in-memory HWPX document)
- Path selection: `mcp-server/src/index.ts:2357`
- Sink: `mcp-server/src/index.ts:2377`
- Sink code: `fs.writeFileSync(tempPath, data);`
- Final write target: `mcp-server/src/index.ts:2441`
- Final target code: `fs.renameSync(tempPath, savePath);`

2. Additional direct write sinks

- Source: `mcp-server/src/index.ts:2156` (`CallToolRequestSchema` handler receives MCP tool arguments)
- Sink: `mcp-server/src/index.ts:3450`
- Sink code: `fs.writeFileSync(outputPath, text, 'utf-8');`

- Sink: `mcp-server/src/index.ts:3486`
- Sink code: `fs.writeFileSync(outputPath, html, 'utf-8');`

3. Related arbitrary read surface

- Source: `mcp-server/src/index.ts:2156` (`CallToolRequestSchema` handler receives MCP tool arguments)
- Sink: `mcp-server/src/index.ts:2326`
- Sink code: `const data = fs.readFileSync(absolutePath);`

8) Attack Prerequisites

- Attacker can invoke the MCP server tools, for example through a configured MCP client, MCP Inspector, or another client connected to the server process.
- The MCP server process has filesystem write permissions to the attacker-selected target path.
- No external sandbox, container policy, or operating-system-level access control prevents the write.

9) Proof of Concept / Reproduction Guidance

This proof of concept demonstrates attacker-controlled file creation using only exposed MCP tools.

1. Create a new in-memory document.

```
{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "tools/call",
  "params": {
    "name": "create_document",
    "arguments": {
      "title": "poc",
      "creator": "poc"
    }
  }
}
```



2. Use the returned `doc_id` to save the document to an attacker-controlled path.

```
{
  "jsonrpc": "2.0",
  "id": 2,
  "method": "tools/call",
  "params": {
    "name": "save_document",
    "arguments": {
      "doc_id": "<doc_id from step 1>",
      "output_path": "/tmp/hwpX-mcp-arbitrary-write.hwpX",

```



```
"create_backup": false,  
"verify_integrity": false  
  }  
}  
}
```

3. Validation

- Confirm that `/tmp/hwpix-mcp-arbitrary-write.hwpix` is created by the MCP server process.
- Repeat with any other path writable by the server process to confirm that `output_path` is not restricted to a safe document directory.

10) Security Impact

- Confidentiality: Low to Medium. The same trust boundary also allows user-selected document paths to be read via `open_document`, depending on parser behavior and file format constraints.
- Integrity: High. An attacker can create or overwrite arbitrary files writable by the MCP server process.
- Availability: High. Overwriting application data, configuration, or other writable files can disrupt the MCP server or host application.
- Scope: Unchanged.

11) CVSS v3.1 Suggestion

- Suggested vector: `CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:H/A:H`
- Suggested base score: 8.1 (High)
- If the MCP server is exposed without authentication or reachable by untrusted clients, consider `PR:N`, which increases severity.

12) Workarounds / Mitigations

- Restrict MCP server access to trusted local users and trusted MCP clients only.
- Run the MCP server under a dedicated low-privilege account with a minimal writable directory.
- Use OS sandboxing, containers, or filesystem permissions to prevent writes outside a designated workspace.
- Avoid passing sensitive filesystem paths to the MCP server until a patch is available.

13) Recommended Fix

- Canonicalize user-supplied paths with `path.resolve` or equivalent and enforce that all output paths remain under an explicit, configured workspace directory.
- Reject absolute paths, parent-directory traversal, symlink escapes, device paths, and other special filesystem targets unless explicitly intended and authorized.
- Add allowlist-based output directories for `save_document`, `export_to_text`, and `export_to_html`.

- Apply the same policy to input paths such as `open_document` and `insert_image` .
- Add regression tests proving that MCP-controlled paths cannot read from or write to locations outside the configured workspace.

14) References

- Repository: <https://github.com/Dayoooun/hwpix-mcp>
- Reviewed source file: `mcp-server/src/index.ts`
- Reviewed source file: `src/mcp/server.ts`
- CWE-73: <https://cwe.mitre.org/data/definitions/73.html>
- CWE-22: <https://cwe.mitre.org/data/definitions/22.html>

15) Credits

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

16) Additional Notes for Form Mapping

- Audit verdict: Exploitable: attacker-controlled MCP tool arguments can reach filesystem write sinks.
- Dynamic exploit replay status: Completed successfully.
- Primary vulnerable tool: `save_document`
- Additional affected tools: `export_to_text` , `export_to_html` , `open_document` , `insert_image` , `insert_image_in_cell`
- Maintainer should validate release mapping before coordinated disclosure.



BruceJqs 2 weeks ago · edited by BruceJqs

Edits ▾

Owner

Author



Tools Q

List Tools

Clear

- get_column_def** >
Get column definition for a section
- set_column_def** >
Set column definition for a section (HWPX only)
- create_document** >
Create a new empty HWPX document
- analyze_xml** >
Analyze document XML for issues like tag imbalance, malformed elements, etc. Useful for diagnosing save failures.
- repair_xml** >
Attempt to repair XML issues in a section. Removes orphan closing tags and fixes table structure.

create_document

Create a new empty HWPX document

Read-only Destructive Idempotent Open-world

title

poc

creator

poc

Tool-specific Metadata: Add Pair

No metadata pairs.

Run Tool

Copy Input

Tool Result: Success

```
{
  doc_id: "3f86glka4"
  format: "hwp"
  message: "New document created"
}
```

Tools Q

List Tools

Clear

- get_tool_guide** >
 START HERE! Get recommended tools for your task. Call this FIRST to understand which tools to use. Available workflows: - "template": Fill content into existing...
- open_document** >
Open an HWPX or HWP document for reading and editing
- close_document** >
Close an open document
- save_document** >
Save the document (HWPX only). Supports backup creation and integrity verification.
- list_open_documents** >
List all currently open documents
- get document text** >

save_document

Save the document (HWPX only). Supports backup creation and integrity verification.

Read-only Destructive Idempotent Open-world

doc_id *

3f86glka4

output_path

/tmp/hwp-mcp-arbitrary-write.hwp

create_backup

Create .bak backup before saving (default: true)

verify_integrity

Verify saved file integrity (default: true)

Tool-specific Metadata: Add Pair

No metadata pairs.

Run Tool

Copy Input

Tool Result: Success

```
{
  message: "Saved to /tmp/hwp-mcp-arbitrary-write.hwp"
  backup_created: true
  integrity_verified: true
}
```


No branches or pull requests

Participants

