

Security Audit Report

security-audit (Claude Code Skill)



Date: 2026-02-06

Auditor: Claude Code (Automated Security Analysis)

Method: Static code analysis, configuration review, dependency analysis

Scope: Full audit (all categories)

Executive Summary

The security-audit project is a Claude Code skill consisting primarily of documentation files (Markdown, HTML template) and a shell installer script. It contains no runtime application code, no dependencies, no Docker configuration, and no CI/CD pipeline. The overall security posture is strong. Two medium-severity findings were identified: an overly permissive Bash execution rule in the local Claude settings file and the use of the `--no-sandbox` flag in the Chrome headless command template. Two low-severity hardening suggestions relate to IDE configuration files committed to the repository and missing `.gitignore` entries for generated output. No critical or high-severity findings were identified.

Results Overview



Findings Overview

ID	SEVERITY	FINDING	FILE	OWASP	CWE
M1	MEDIUM	Overly Broad Bash Permission in Local Settings	<code>.claude/settings.local.json</code>	A05:2021	CWE-269
M2	MEDIUM	--no-sandbox Flag in Chrome Headless Command	<code>SKILL.md</code>	A05:2021	CWE-693
L1	LOW	IDE Configuration Committed to Repository	<code>.idea/</code>	A05:2021	CWE-538
L2	LOW	Missing Output Directory	<code>.gitignore</code>	-	CWE-538

ID	SEVERITY	FINDING	FILE	OWASP	CWE
		Exclusions in .gitignore			

Detailed Findings

M1 – Overly Broad Bash Permission in Local Settings

MEDIUM

Description: The local Claude Code settings file contains the permission rule `Bash(bash:*)`, which grants unrestricted bash command execution without user confirmation. This violates the principle of least privilege. If other skills or prompts are executed within this project context, they could leverage this broad permission to execute arbitrary commands without the user being prompted for approval.

Affected Code

File: `.claude/settings.local.json`, Lines 3-7

```
"permissions": {  
    "allow": [  
        "Bash(git add:*)",  
        "Bash(git commit:*)",  
        "Bash(bash:*)"  
    ]  
}
```

Recommended Fix

```
"permissions": {  
    "allow": [  
        "Bash(git add:*)",  
        "Bash(git commit:*)"  
    ]  
}
```

OWASP: A05:2021 Security Misconfiguration

CWE: CWE-269 Improper Privilege Management

M2 – --no-sandbox Flag in Chrome Headless Command

MEDIUM

Description: The Chrome headless command in the skill prompt uses `--no-sandbox`, which disables Chrome's sandbox security. While this flag is sometimes needed in Docker containers running as root, it is unnecessary and insecure on local macOS execution where the skill is designed to run. If the HTML file being rendered were to contain malicious content, the lack of sandbox would increase the attack surface.

Affected Code

File: `SKILL.md`, Lines 147-151

```
"/Applications/Google Chrome.app/Contents/MacOS/Google Chrome" \
--headless --disable-gpu --no-sandbox \
--print-to-pdf="docs/security-audit/[projectname]-security-audit-YYYY-MM-
DD.pdf" \
--print-to-pdf-no-header \
"docs/security-audit/[projectname]-security-audit-YYYY-MM-DD.html"
```

Recommended Fix

```
"/Applications/Google Chrome.app/Contents/MacOS/Google Chrome" \
--headless --disable-gpu \
--print-to-pdf="docs/security-audit/[projectname]-security-audit-YYYY-MM-
DD.pdf" \
--print-to-pdf-no-header \
"docs/security-audit/[projectname]-security-audit-YYYY-MM-DD.html"
```

OWASP: A05:2021 Security Misconfiguration CWE: CWE-693 Protection Mechanism Failure

L1 – IDE Configuration Committed to Repository

LOW

Description: The `.idea/` directory containing JetBrains IDE configuration files is tracked in git, despite being listed in `.gitignore`. The `workspace.xml` file exposes local environment details including the PHP interpreter version, project IDs, and internal metadata. While not directly exploitable, this constitutes unnecessary information disclosure.

Affected Code

Directory: `.idea/` (`workspace.xml`, `misc.xml`, `php.xml`, `vcs.xml`)

```
# .idea/workspace.xml excerpt:  
<component name="PhpWorkspaceProjectConfiguration" interpreter_name="PHP 8.3" />  
<component name="ProjectId" id="39I4j9m7HTo61wS2lknQLwY04jr" />
```

Recommended Fix

```
# Remove from git tracking (the .gitignore rule already exists):  
git rm -r --cached .idea/
```

OWASP: A05:2021 Security Misconfiguration

CWE: CWE-538 Insertion of Sensitive Information into Externally-Accessible File or Directory

L2 – Missing Output Directory Exclusions in .gitignore

LOW

Description: The `.gitignore` file excludes `*.pdf` and `*.html` (except the template), but does not exclude the `docs/epics/` directory where generated Markdown epic files are written. When the skill is run against its own repository, the generated epic Markdown files would appear as untracked files and could be accidentally committed.

Affected Code

File: `.gitignore`, Lines 1-7

```
*.pdf  
*.html  
!report-template.html  
node_modules/  
.DS_Store  
.idea/  
.claude/
```

Recommended Fix

```
*.pdf  
*.html  
!report-template.html  
node_modules/  
.DS_Store  
.idea/  
.claude/  
docs/epics/  
docs/security-audit/
```

CWE: CWE-538 Insertion of Sensitive Information into Externally-Accessible File or Directory

Positive Findings

No Hardcoded Secrets or Credentials: The codebase contains no API keys, tokens, passwords, or other sensitive credentials. All files are documentation or configuration only.

Robust .gitignore Configuration: Generated output files (PDF, HTML) are properly excluded from version control, with an explicit exception for the report template.

Safe Bash Defaults in Install Script: The `install.sh` script uses `set -euo pipefail`, ensuring that the script exits on errors, unset variables, and pipe failures.

Install Script Path Validation: The installer properly checks if the target path exists, differentiates between symlinks and regular files, and handles update vs. fresh install scenarios safely.

Zero Runtime Dependencies: The project has no `package.json`, `requirements.txt`, or similar dependency manifests. This eliminates the entire class of supply-chain vulnerabilities.

Symlink-Based Installation: The installation uses symbolic links rather than file copies, ensuring updates are immediately reflected without re-installation.

Risk Matrix

	LOW (IMPACT)	MEDIUM (IMPACT)	HIGH (IMPACT)	CRITICAL (IMPACT)
High (Likelihood)				
Medium (Likelihood)	L1, L2	M1		
Low (Likelihood)		M2		

References

OWASP Top 10:2021: <https://owasp.org/Top10/>

OWASP A05:2021 – Security Misconfiguration: https://owasp.org/Top10/A05_2021-Security_Misconfiguration/

CWE-269 – Improper Privilege Management: <https://cwe.mitre.org/data/definitions/269.html>

CWE-538 – Insertion of Sensitive Information into Externally-Accessible File or Directory: <https://cwe.mitre.org/data/definitions/538.html>

CWE-693 – Protection Mechanism Failure: <https://cwe.mitre.org/data/definitions/693.html>

OWASP Cheat Sheet Series: <https://cheatsheetseries.owasp.org/>

CWE (Common Weakness Enumeration): <https://cwe.mitre.org/>

NIST NVD (National Vulnerability Database): <https://nvd.nist.gov/>

Docker Security Best Practices: <https://docs.docker.com/engine/security/>

OWASP Docker Security Cheat Sheet:

https://cheatsheetseries.owasp.org/cheatsheets/Docker_Security_Cheat_Sheet.html

This report was generated automatically using the [Claude Code Security Audit Skill](#).

The analysis is based on static code analysis and may not cover all vulnerabilities.

A manual review by a security expert is recommended.