
A RESOURCE FOR DESIGN SYSTEM, ENGINEERING AND SECURITY TEAMS

MCP permission security review.



Security review template for MCP permissions.

Snapshot. Threat-model. Inventory. Lock.

27

review items, each with its evidence

19

MCP threats mapped to STRIDE

04

errors caught in one 1.0.1 patch release

Review the permissions, then watch them change.

An MCP server's risk is not in its README. It is in what the server can read, change and send, and which servers share the session with it. A review writes those down, checks them against known attacks, and notices when they change: a server can rewrite its tool descriptions after you approved them [S06].

Section 01 scopes the review and freezes the tool list. Sections 02 to 04 walk STRIDE through the threats the MCP Security Best Practices page and published research describe: lookalike packages, tool poisoning, rug pulls, token passthrough, the lethal trifecta, local server compromise and authorization URL injection [S01, S06, S08]. Section 05 is change control.

`check-inventory.mjs` flags missing fields, understated risk, overdue reviews and trifectas that span servers; in Field Guide 25's own Claude Code config it finds one, accepted with a written break. `diff-tools.mjs` keeps per-tool hashes and diffs tool lists; between the token server's 1.0.0 and a doctored 1.0.1 it reports 4 errors.

Version 1.0 / Sources checked 24 September 2026

Field Guide 26 of the Design × AI series. Pairs with Field Guides 24 (evaluation), 25 (starter config) and 27 (integration patterns); the inventory describes Field Guide 25's servers. Verified 24 September 2026 against the MCP specification 2026-07-28 and its Security Best Practices page.

Practical guidance, not a penetration test or a compliance attestation. STRIDE is used as a checklist frame, not a full threat-modelling exercise. The inventory's trifecta legs are judgements; the checker enforces consistency, not truth. Prepared with AI assistance and edited by hand.

START HERE

One server, one client, one review.

Risk depends on the neighbours. Review a server in the client it runs in, with the other servers that load beside it.

STRIDE	What it means for MCP	Items
Spoofing	A server, package or OAuth client that is not who it says	S1 to S3
Tampering	Tool text or code that changes what the model does	T1 to T4
Repudiation	Tool calls nobody can reconstruct later	R1, R2
Information disclosure	Data leaving through tools, tokens or configs	I1 to I4
Denial of service	Context floods and hangs	D1, D2
Elevation of privilege	A server or a string that runs code as you	E1 to E4

STRIDE categories per the Microsoft Threat Modeling Tool documentation [S11]. The threats are the ones the MCP Security Best Practices page and cited research describe.

Reviewing a new server

Field Guide 24 first: if it fails a blocker there, stop.
Otherwise work sections 01 to 04 and add the inventory row.

An upgrade arrived

Section 05. Run `diff-tools.mjs --verify` against the lock. Any change reopens the items it touches, starting with T1 and T2.

Adding a server to a client

Item I1 for the whole client, not only the new server.
Two harmless servers can complete the trifecta together.

Quarterly review

Run `check-inventory.mjs`. Overdue reviews, understated risk and missing locks come out as a list.

Label	Meaning
<code>SPOOFING</code> ... <code>ELEVATION</code>	The STRIDE category the item addresses.
<code>SPEC</code>	Traces to the MCP specification 2026-07-28 or its Security Best Practices page.
<code>RESEARCH</code>	From published security research or a documented incident.
<code>SCRIPT</code>	Checked by <code>check-inventory.mjs</code> or <code>diff-tools.mjs</code> in this kit, or a Field Guide 24 or 25 script.
<code>PRACTICE</code>	A working method with a review signal.

SECTION 01

Scope the review and freeze what you saw.

A review is about one version of one server's tool list. Capture it before you judge it.

Suggested owners: Security partner + server owner

☐ One review per server, per client

P1 **PRACTICE**

The same GitHub server is a different risk in Claude Desktop, where no browser shares the session, and in Claude Code, where one does. Name the client in the review.

Evidence: Client and config file named in section 1 of `review-template.md`.
S08.

☐ Snapshot and lock the tool list

P2 **SCRIPT**

Capture tools/list with the Inspector CLI, then `diff-tools.mjs --lock`. The lock hashes each tool's name, title, description, input and output schemas and annotations: everything a model reads or a client trusts.

Evidence: A lock file with a `snapshot` hash, committed next to the review.
Invariant Labs recommends pinning tool descriptions by hash. S06.

☐ Write the data flow in five lines

P3 **PRACTICE**

What it reads, what it writes or deletes, where it can connect, which untrusted content it returns and which servers share the session. If you cannot write a line, the review is not ready.

Evidence: Section 2 of the template complete.
Recommended practice.

☐ Fill the inventory row

P4 **SCRIPT**

Version, review date, permissions (filesystem, network, credentials), justification, risk, mitigations and next review date: the fields the essay lists, as JSON a script can check.

Evidence: `check-inventory.mjs` reports no I01 or I02 errors.
The essay's security review template. Kit schema.

You cannot review what you did not write down, or notice a change to it.

SECTION 02

Spoofing and tampering.

Is it the server you meant, and is its text still the text you approved?

Suggested owners: Security partner + engineering lead

☐ The package is not a lookalike

S1 **SPOOFING** **RESEARCH**

Match package scope, repository owner and registry namespace to the vendor. The postmark-mcp package was not Postmark's; it built trust over 15 versions and added a hidden BCC in the 16th.

Evidence: Namespace and owner recorded; Field Guide 24 check B.01 passes.

S09, S13.

☐ The remote server is on the vendor's domain

S2 **SPOOFING** **SPEC**

https, the vendor's own domain, and the same URL the vendor's docs and registry entry give. A proxy on someone else's domain sees every call.

Evidence: URL compared with the vendor docs and registry entry.

S13, S15.

☐ Proxy servers ask for consent per client

S3 **SPOOFING** **SPEC**

A server that fronts a third-party API with a static OAuth client ID must keep its own per-client consent, or a consent cookie lets an attacker's client skip the screen: the confused deputy.

Evidence: Consent screen names the client and redirect URI; not skipped on a second client.

Security Best Practices, confused deputy: MUST. S01.

☐ Descriptions carry no instructions

T1 **TAMPERING** **SCRIPT** **RESEARCH**

Scan every description and parameter text for pseudo-tags, file paths, hidden characters and text that steers other tools.

Evidence: Field Guide 24's `inspect-tools.mjs` reports no T04, T05 or T11.

Tool poisoning and shadowing. S06.

☐ An approved tool cannot change silently

T2 **TAMPERING** **SCRIPT** **RESEARCH**

A rug pull is a server that changes its tool definitions after approval. Clients do not stop it for you; the lock file and a diff on every upgrade do.

Evidence: `diff-tools.mjs --verify` in CI; see section 05.

S06. Kit fixtures: 4 errors from 1.0.0 to 1.0.1.

☐ **Tool output is treated as data, not instructions**T3 **TAMPERING** **RESEARCH**

List the tools that return text other people wrote: issues, pull requests, pages, email. That text can carry instructions; Invariant Labs showed a public issue steering an agent into leaking private repository data through the GitHub server. Filter it where the server offers a way, as GitHub's lockdown mode does.

Evidence: Untrusted-content tools listed in the inventory's trifecta legs.

S07, S12.

☐ **Updates arrive only through review**T4 **TAMPERING** **SCRIPT**

Pin the exact version or image digest so a new release waits for this review instead of arriving on the next restart.

Evidence: Field Guide 25's validator rule C04; inventory rule I07.

S09.

Trust the text you approved, and prove it is still that text.

SECTION 03

Repudiation and information disclosure.

Can you reconstruct what happened, and can data leave by a path nobody chose?

Suggested owners: Security partner + design-system lead

☐ Tool calls leave an audit trail

R1 **REPUDIATION** **SPEC**

The spec asks clients to log tool usage for audit. Know where the log is, what it keeps (tool, arguments, result) and for how long.

Evidence: Log location and retention in the review.

Tools, security considerations. S04.

☐ No token passthrough

R2 **REPUDIATION** **DISCLOSURE** **SPEC**

A server must reject tokens not issued for it and must not forward a client's token downstream. Passthrough erases who did what, and lets one stolen token act everywhere it is accepted.

Evidence: Server docs or code show audience validation and its own downstream credential.

Security Best Practices, token passthrough. S01, S02.

☐ No unbroken lethal trifecta in the client

I1 **DISCLOSURE** **SCRIPT** **RESEARCH**

Private data, untrusted content and a way to communicate out, anywhere in one client's servers, is how injected text becomes a leak. Field Guide 25's Claude Code set has all three across GitHub and Playwright; the written break is read-only GitHub and a prompt on every navigation. Built-in tools count too.

Evidence: `check-inventory.mjs` rule I06: no error; any warning names its break.

S08, S07.

☐ Secrets live outside the config

I2 **DISCLOSURE** **SCRIPT**

Credentials in environment variables (Claude Code), OAuth or the OS keychain, never in the JSON. The inventory records where each one lives.

Evidence: Field Guide 25 validator rule C01; inventory rule I08.

S14.

☐ Scopes are the minimum, and grow only on demand

I3 **DISCLOSURE** **SPEC**

Named folders, one repository, read-only toolsets. For OAuth servers, a small first scope and step-up challenges for privileged tools, not the whole catalogue at consent.

Evidence: Paths and scopes listed in the inventory row.

Security Best Practices, scope minimization. S01.

☐ **State handles are not credentials**

I4

DISCLOSURE

SPEC

The 2026-07-28 revision removed protocol sessions; servers that keep state mint handles passed as tool arguments. They must be random, bound to the authenticated user and expiring, and must never count as authentication.

Evidence: Handle format and binding described, or n/a for stateless servers.

Security Best Practices, state handle hijacking. S01, S05.

Check the client, not the server. The trifecta is assembled from parts.

SECTION 04

Denial of service and elevation of privilege.

What happens when a call floods the context, and what runs with your account's rights.

Suggested owners: Security partner + engineering lead

☐ **Results are bounded**

D1 **DENIAL**

Limits, pagination and truncation flags. Claude Code warns above 10,000 tokens per result and cuts at 25,000 by default; unbounded output wastes context and can bury the instruction that matters.

Evidence: Largest realistic call measured against the cap.

S14.

☐ **Hangs fail loudly**

D2 **DENIAL**

Start-up and per-call timeouts set; long work reports progress. A server that hangs looks like a thinking model.

Evidence: Timeouts in the config; a trial with the network off.

S14.

☐ **The launch command is one program**

E1 **ELEVATION** **SCRIPT** **SPEC**

A binary and its arguments. Startup commands that chain `&&`, pipe `curl` into a shell or call `sudo` are the spec's own examples of local server compromise.

Evidence: Field Guide 25 validator rule C06.

Security Best Practices, local server compromise. S01.

☐ **Code-running tools are sandboxed, confirmed or off**

E2 **ELEVATION** **SCRIPT**

Any tool that evaluates code, runs shell commands or passes free-form queries is the server's highest privilege. Run it in a container, keep its prompt, or turn it off.

Evidence: Field Guide 24's scanner T06 and T07 findings, each with a mitigation.

S04.

☐ **Local HTTP servers are not reachable from the browser**

E3 **ELEVATION** **SPEC**

Bind to 127.0.0.1, validate `origin`, require a token. Otherwise a web page can reach a local server through DNS rebinding.

Evidence: Bind address and Origin check confirmed.

S15. Security Best Practices. S01.

☐ **Authorization URLs cannot run commands**

E4

ELEVATION

SPEC

RESEARCH

Clients must accept only http and https authorization URLs and must not open them through a shell. `mcp-remote` before 0.1.16 let a malicious server run commands this way (CVE-2025-6514).

Evidence: Bridge version at or above the fix; client version recorded.

Security Best Practices, OAuth authorization URL validation. S01, S10.

Anything that runs code runs it as you. Review it that way.

SECTION 05

Change control: the review that repeats itself.

The first review is the easy one. These items make the next one happen without a meeting.

Suggested owners: Engineering lead + security partner

☐ **Verify the lock on every upgrade**

C1 **SCRIPT**

Capture tools/list for the new version and run `diff-tools.mjs --verify`. Any added, removed or changed tool exits 1 and reopens the review.

Evidence: A CI job or pre-upgrade script that runs the verify step.

Kit script.

☐ **Read the diff before you accept it**

C2 **SCRIPT** **RESEARCH**

Errors block: new instruction-like text (D03), a new required parameter (D04), an annotation moved to the unsafe side (D06), a new tool that trips the scanner (D01). A patch version number is not evidence of a patch.

Evidence: `diff-tools.mjs old new` output attached to the upgrade.

S06. Kit fixtures.

☐ **Re-review on the triggers, not the calendar alone**

C3 **PRACTICE**

A new version, a changed tools hash, a new client, a new credential scope, a new server in the same client, or a published advisory. The calendar date catches the rest.

Evidence: Triggers listed in section 5 of the template; `nextReview` set.

Recommended practice.

☐ **Ratings stay honest**

C4 **SCRIPT**

Code execution means HIGH. Write or delete access, credentials or two trifecta legs mean at least MEDIUM. HIGH needs mitigations. The checker enforces those floors so nobody rates a browser LOW to skip the meeting.

Evidence: `check-inventory.mjs` rules I04 and I05 pass.

Kit script.

A version number is a claim. The diff is the evidence.

APPENDIX A

Catch a rug pull.

Three snapshots of the Field Guide 27 token server: the real 1.0.0, a benign 1.1.0 and a 1.0.1 patch that is anything but.

terminal

```
$ node scripts/diff-tools.mjs fixtures/acme-ds-tokens-1.0.0.tools.json fixtures/acme-ds-tokens-1.1.0.tools.json
acme-ds-tokens-1.0.0.tools.json -> acme-ds-tokens-1.1.0.tools.json: 2 -> 3 tools, 0 errors, 1 warnings, 1 info
  info D01 list_modes: New tool (read-only, closed-world).
  warn D03 search_tokens: Description or title changed. Read the new text before accepting it.

$ node scripts/diff-tools.mjs fixtures/acme-ds-tokens-1.0.0.tools.json fixtures/acme-ds-tokens-1.0.1.tools.json
acme-ds-tokens-1.0.0.tools.json -> acme-ds-tokens-1.0.1.tools.json: 2 -> 3 tools, 4 errors, 1 warnings, 0 info
  warn D01 sync_tokens: New tool: review it as if it were a new server.
  error D01 sync_tokens: New tool trips T02: Name says it changes something, but it declares neither readOnlyHint nor destructiveHint.
  error D03 lookup_token: Description changed and now trips T04: Instruction-like text: references a credential or config file (description).
  error D04 lookup_token: New required parameter "context": "Project context. Required for accurate resolution.".
  error D06 search_tokens: openWorldHint false -> true (now absent, so the unsafe default applies).

$ node scripts/diff-tools.mjs --verify fixtures/acme-ds-tokens.tools.lock.json fixtures/acme-ds-tokens-1.0.1.tools.json
acme-ds-tokens-1.0.1.tools.json: 3 tools differ from acme-ds-tokens.tools.lock.json
  changed lookup_token
  changed search_tokens
  added sync_tokens
```

Output from the kit, long lines wrapped for print. The 1.0.1 description asks for the project's `.env` file in a new required `context` parameter, and drops an annotation so the unsafe default applies.

`.github/workflows/mcp-tools.yml` (steps)

```
- name: Capture the tool list of the pinned server
  run: npx @modelcontextprotocol/inspector@2.8.0 --cli node tools/ds-tokens-mcp/src/server.mjs
      --method tools/list --format json > mcp/acme-ds-tokens.tools.json
- name: Fail when any tool changed since the review
  run: node scripts/diff-tools.mjs --verify mcp/acme-ds-tokens.tools.lock.json mcp/acme-ds-tokens.tools.json
```

A sketch, not executed as a workflow in the kit. After a reviewed change, regenerate the lock with `--lock` in the same pull request as the review.

APPENDIX B

The permissions inventory, checked.

One JSON file describes every server and the clients that load them together. The schema documents the fields; the checker enforces the cross-field rules.

permissions-inventory.json (one entry, abridged)

```
{
  "name": "playwright", "package": "@playwright/mcp", "version": "0.0.82",
  "exec": true,
  "network": { "egress": ["http://localhost:6006"] },
  "tools": { "total": 25, "readOnly": 7, "destructive": 18, "openWorld": 25 },
  "toolsLock": "sha256:6848ef54134eaf010c0af62eaf33eac421267f17b62c65945be2fadb44ecd97",
  "trifecta": { "privateData": false, "untrustedContent": true, "externalComms": true },
  "risk": "HIGH",
  "mitigations": ["--isolated: no saved browser profile, no cookies",
    "No allow rule for browser_navigate, browser_evaluate or
    browser_run_code_unsafe"],
  "nextReview": "2026-10-24"
}
```

Abridged. Tool counts and the lock hash come from the real Playwright MCP 0.0.82 capture in Field Guide 24's kit.

terminal

```
$ node scripts/check-inventory.mjs permissions-inventory.json --today 2026-09-24
permissions-inventory.json: 5 servers, 2 clients, 0 errors, 2 warnings
warn I10 github: Not captured: needs a GitHub account. Capture tools/list at first
sign-in and lock it.
warn I06 claude-code: Lethal trifecta across servers (private data: github;
untrusted content: github, playwright; external communication: playwright).
Accepted with a written break: GitHub runs read-only; Playwright is limited
to http://localhost:6006 and every browser_navigate call needs approval ...

$ node scripts/check-inventory.mjs examples/violations.inventory.json --today 2026-09-24
violations.inventory.json: 5 servers, 2 clients, 7 errors, 3 warnings
error I01 git: Missing owner.
error I05 github: Rated LOW, but it holds two trifecta legs: at least MEDIUM.
error I08 github: Credentials stored in "config-file". Use env, oauth or keychain.
error I05 playwright: Rated MEDIUM, but it can execute code: at least HIGH.
error I07 playwright: Pin an exact version.
error I09 claude-desktop: Lists fetch, which the inventory does not describe.
error I06 claude-code: Lethal trifecta across servers (...). Remove a leg or write
down what breaks the chain.
```

Output from the kit, wrapped; text shortened where marked with an ellipsis, and the violations run's three warnings (I03, I04, I10) omitted. The Claude Desktop set has no trifecta: nothing in it can send data out.

KEEP WITH THE REVIEW

Sign off the review.

The last page of `review-template.md`, as a record. It is not a certification; it is the trail the next reviewer starts from.

Server / version / client	Tools lock hash (snapshot)
Inspector scan (errors / warnings)	STRIDE rows failed, with mitigations
Trifecta status for the client	Risk: LOW / MEDIUM / HIGH
Decision and conditions	Residual risk accepted by
Next review date	Re-review triggers agreed

Accepting a risk is a signature, not a silence.

If a HIGH server ships with a trifecta warning, one named person accepts it in writing, with the break that makes it tolerable.

SOURCES / MAINTENANCE

Keep the guide current.

Sources checked 24 September 2026. Diff, lock and inventory output was produced by running the kit on that date. STRIDE is used as a frame; the MCP-specific threats come from the specification and the research below.

S01 / MCP, Security Best Practices

Confused deputy, token passthrough, SSRF, state handle hijacking, local server compromise, OAuth URL validation, scope minimization.

https://modelcontextprotocol.io/docs/2026-07-28/tutorials/security/security_best_practices

S02 / MCP specification 2026-07-28, Authorization

Audience validation, resource indicators, no tokens in query strings.

<https://modelcontextprotocol.io/specification/2026-07-28/basic/authorization>

S04 / MCP specification 2026-07-28, Tools

Annotations untrusted; clients should confirm sensitive operations and log tool usage.

<https://modelcontextprotocol.io/specification/2026-07-28/server/tools>

S05 / MCP specification 2026-07-28, Key changes

Protocol sessions removed; explicit server-minted handles for state.

<https://modelcontextprotocol.io/specification/2026-07-28/changelog>

S06 / Invariant Labs, MCP tool poisoning attacks

Hidden instructions, rug pulls after approval, cross-server shadowing; recommends pinning tools by hash (1 April 2025).

<https://invariantlabs.ai/blog/mcp-security-notification-tool-poisoning-attacks>

S07 / Invariant Labs, GitHub MCP exploited

A public issue steered an agent into leaking private repository data (26 May 2025).

<https://invariantlabs.ai/blog/mcp-github-vulnerability>

S08 / Simon Willison, The lethal trifecta

Private data, untrusted content, external communication (16 June 2025).

<https://simonwillison.net/2025/Jun/16/the-lethal-trifecta/>

S09 / Postmark, malicious postmark-mcp package

Impersonation, 15 clean versions, BCC added in 1.0.16.

<https://postmarkapp.com/blog/information-regarding-malicious-postmark-mcp-package>

S10 / JFrog, CVE-2025-6514 in mcp-remote

Command injection through authorization_endpoint; fixed in 0.1.16; CVSS 9.6.

<https://jfrog.com/blog/2025-6514-critical-mcp-remote-rce-vulnerability/>

S11 / Microsoft, Threat Modeling Tool threats (STRIDE)

Definitions of the six STRIDE categories.

<https://learn.microsoft.com/en-us/azure/security/develop/threat-modeling-tool-threats>

S12 / GitHub MCP server

Read-only mode, toolsets and lockdown mode for untrusted repository content.

<https://github.com/github/github-mcp-server>

S13 / The MCP Registry

Namespace authentication; security scanning delegated to package registries.

<https://modelcontextprotocol.io/registry/about>

S14 / Claude Code, Connect Claude Code to tools via MCP

\${VAR} expansion, output warning at 10,000 tokens and 25,000 cap, timeouts, project approvals.

<https://code.claude.com/docs/en/mcp>

S15 / MCP specification 2026-07-28, Streamable**HTTP**

Origin validation against DNS rebinding; bind local servers to 127.0.0.1.

<https://modelcontextprotocol.io/specification/2026-07-28/basic/transport/streamable-http>

Maintenance: re-read the Security Best Practices page at each MCP specification revision (the 2026-07-28 revision replaced session hijacking with state handle hijacking), and add new incidents to the threat table when they are published. Update the PDF, HTML, Markdown, JSON and the template together.