{"session_id": "A7f3e2B9d1C45aB8",
"user_id": "admin_01",
"cookies": {
"auth_token": "xJ8kL9mPqR2sT5v
"session_key": "v4n2p6q8r1t3z5
"mcp_access": " true",
"recommended_plugins": [
"community_server_A",
"legacy_connector_B"]
},
"timestamp": "2024-05-22T14:30:00Z
}

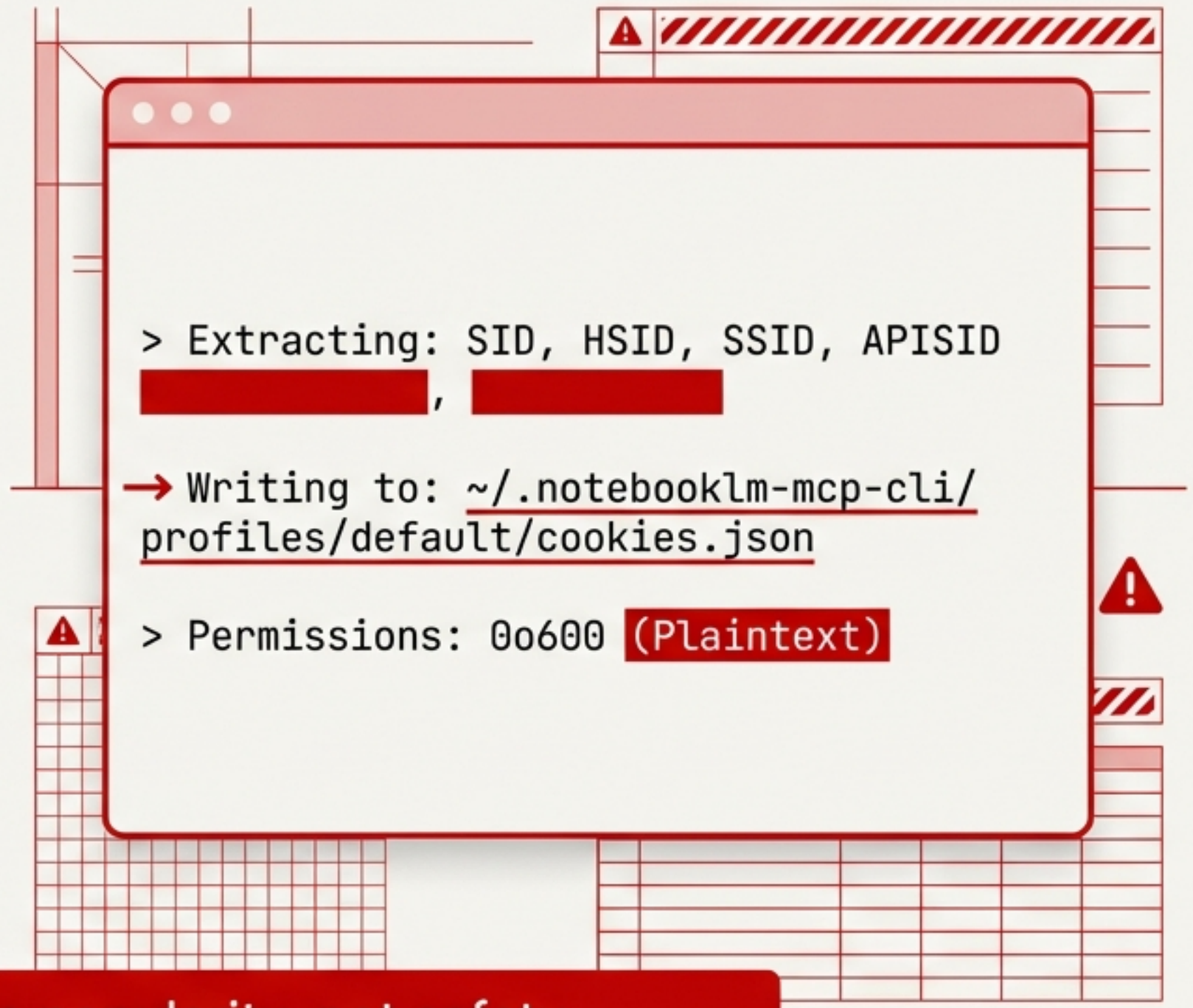
The Scariest Vulnerabilities Come Recommended.

A Security Case Study on the Hidden Risks of Community MCP Servers.

Threat Intelligence Blueprint & Architectural Post-Mortem

The Illusion

The Reality



Core Insight: Stars measure popularity, not safety.
AI assistants optimize for functional match, not security posture.

Sr. AppSec Engineer (Lead)

Focus: Overall architecture, auth model, go/no-go verdict.

Pentester

`>_` `> exploit`

Focus: File system access, injection vectors, network exposure.

Target: notebooklm-mcp-cli

Security Engineer

`sid, hsid`

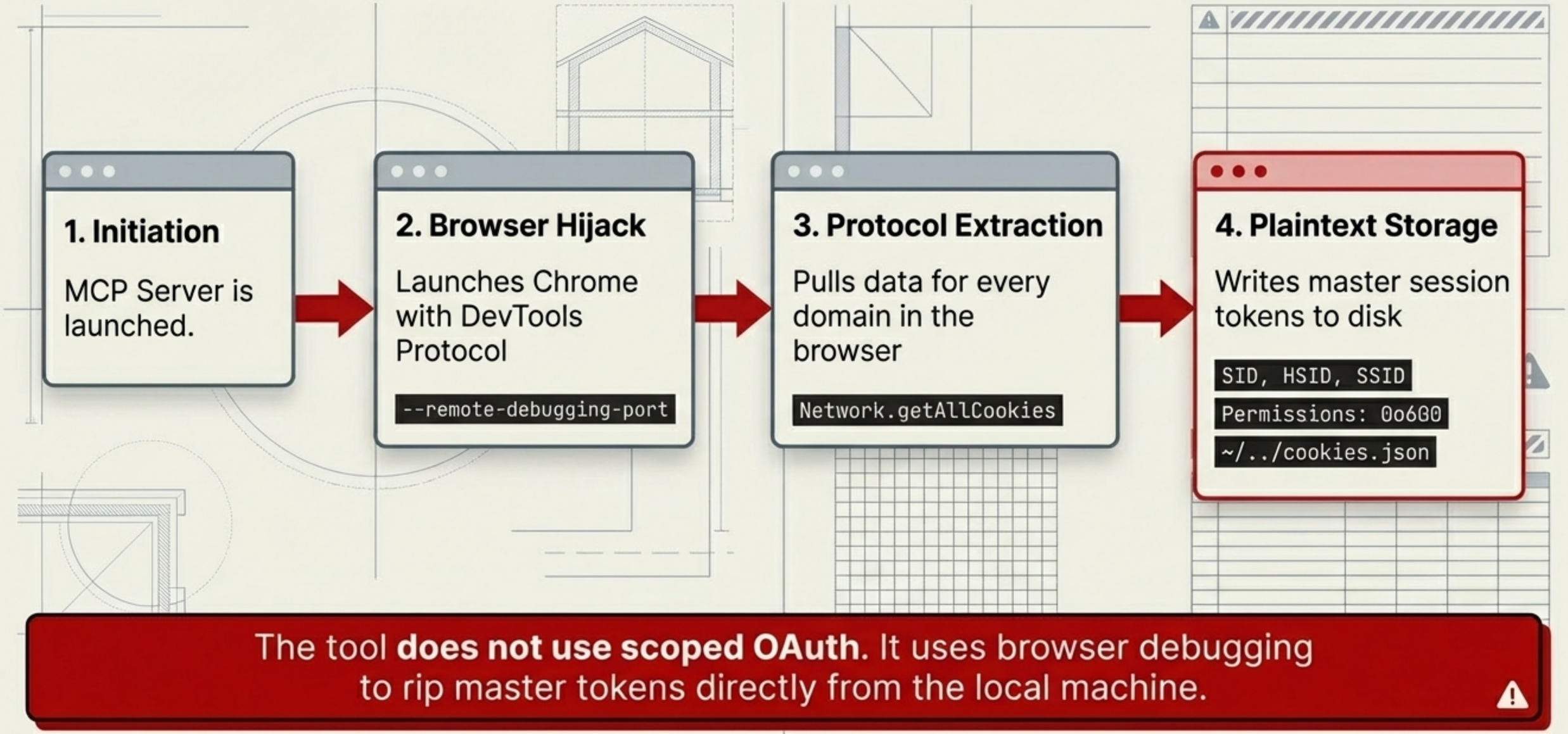
Focus: Cookie scope, session replay, ToS compliance.

Developer

Focus: Workflow fit, dependency quality, installation friction.

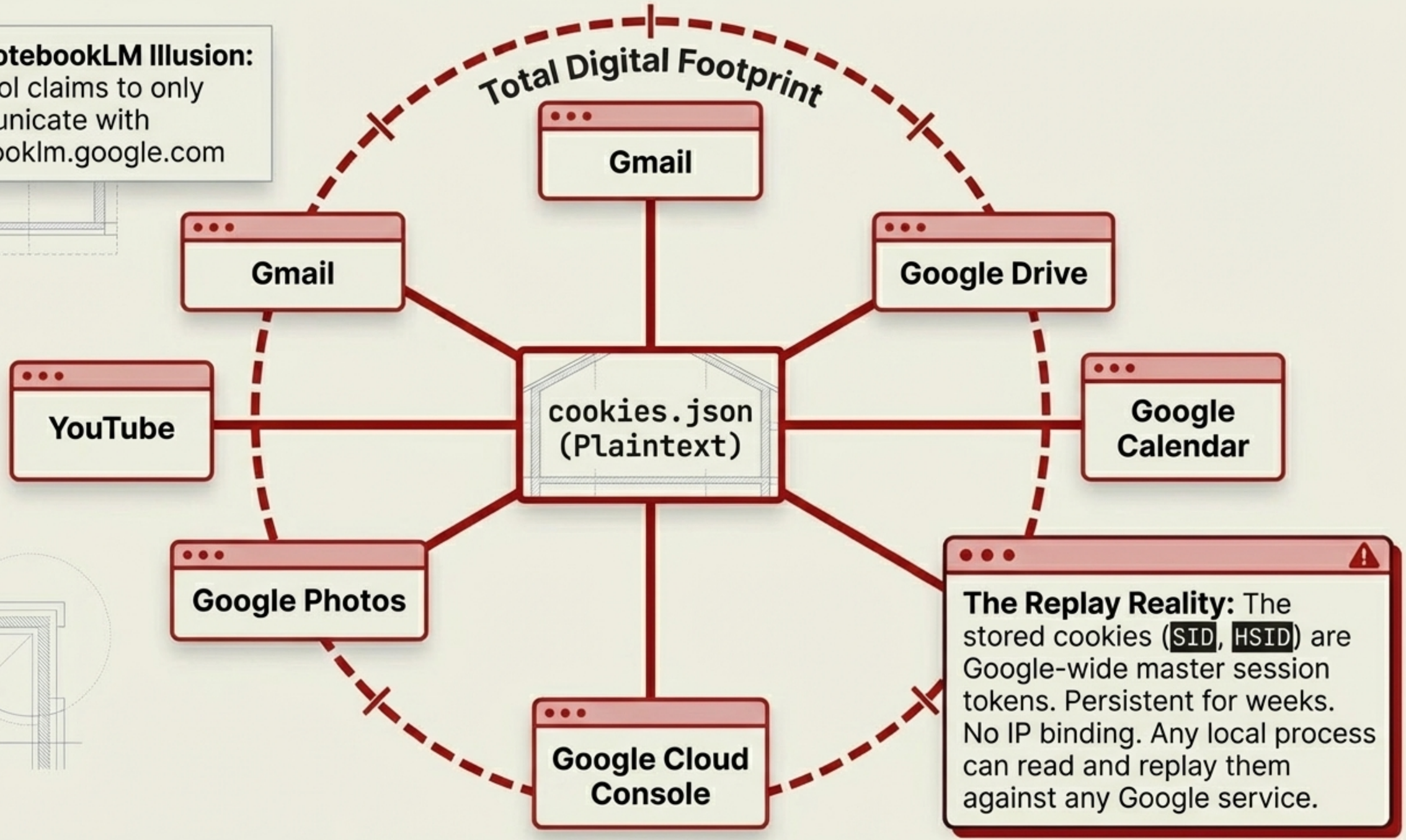
Yielded **11 distinct security findings**. Running parallel agents provides a 360-degree assessment. Convergence across distinct personas provides a high-confidence threat signal.

Editorial AppSec Audit Report



The tool **does not use scoped OAuth**. It uses browser debugging to rip master tokens directly from the local machine.

The NotebookLM Illusion:
The tool claims to only communicate with notebooklm.google.com



Editorial AppSec Audit Report

Threat Dashboard

ToS Violation & Account Ban Risk

Reverse-engineering of internal RPCs and spoofing of Chrome 143.

`batchexecute`

⚠ Violates ToS; risks immediate, unappealable Google account suspension.

Massive Prompt Injection Surface

29 exposed MCP tools.

⚠ Every tool exposed to the AI (e.g., notebook_share_public) is a potential action a malicious prompt can trigger.

Network Exposure

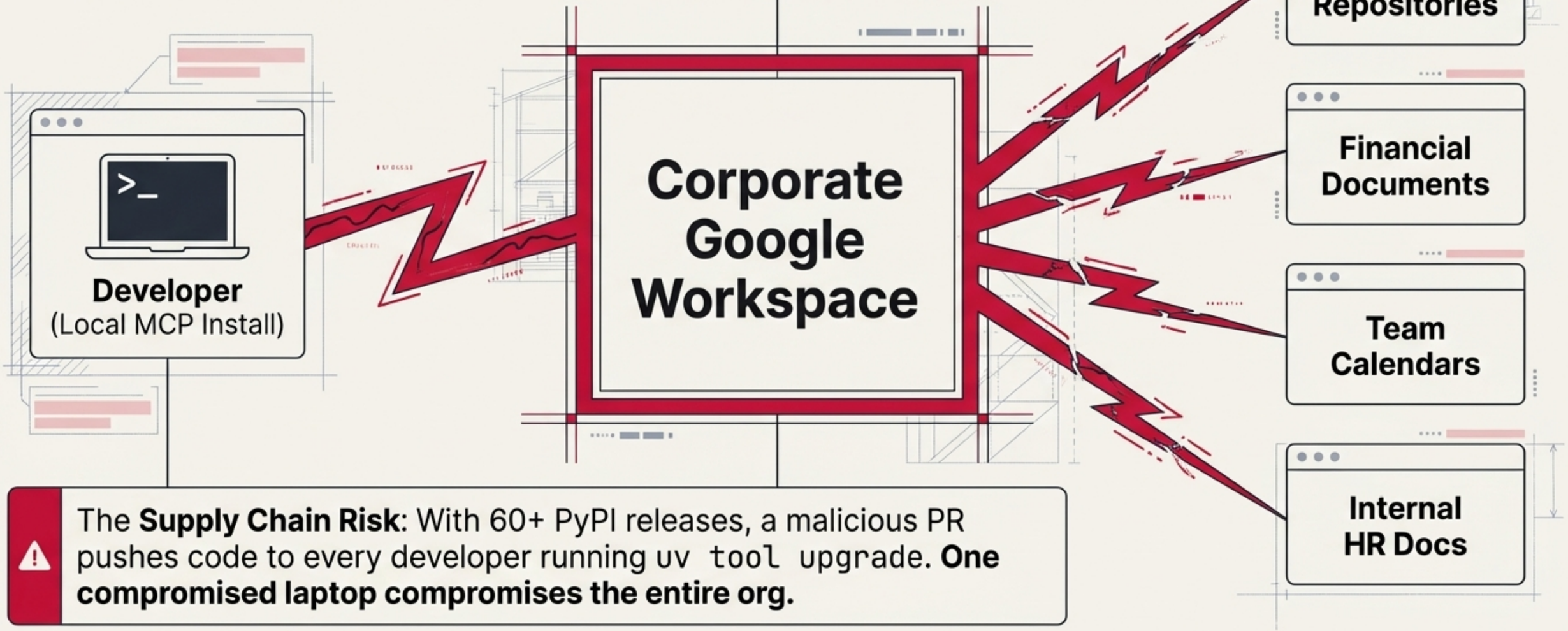
Chrome DevTools debugging port binds to localhost during login.

`--remote-allow-origins=*`

⚠ Allows arbitrary JavaScript execution by other local processes during the login window.

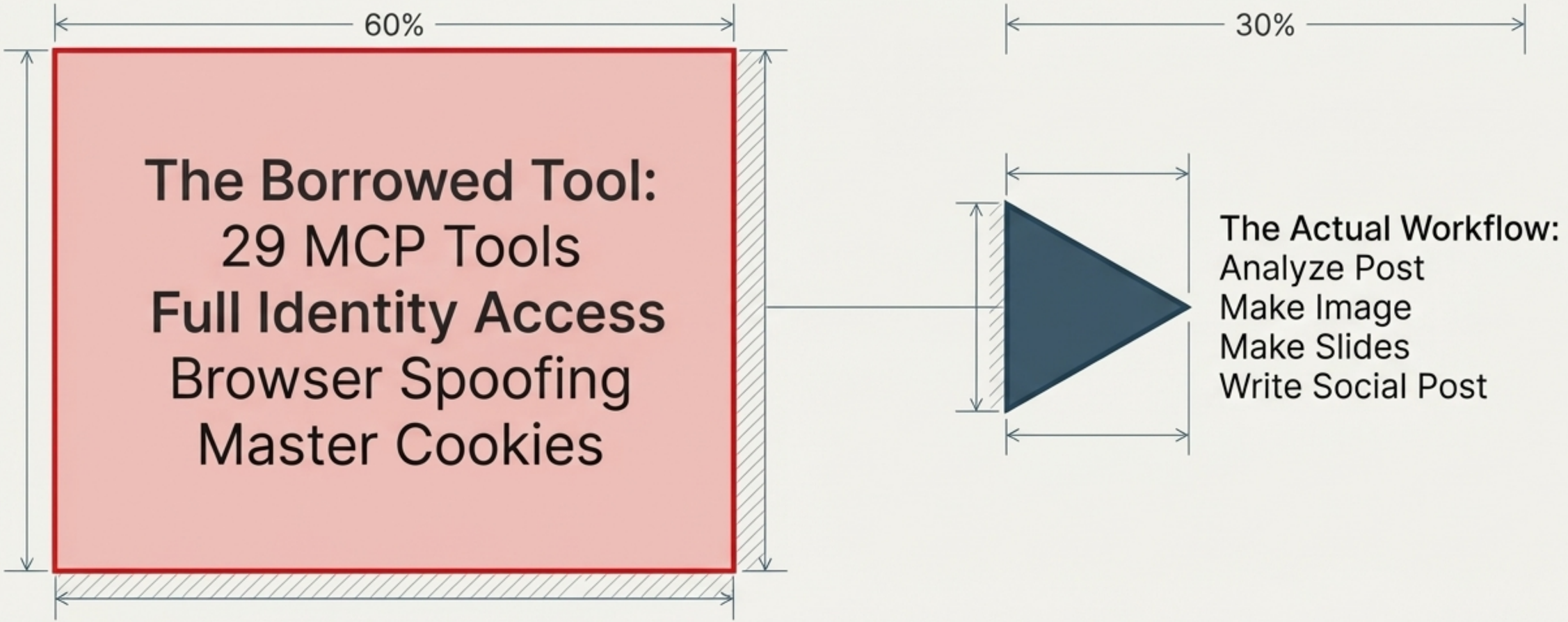
The Governance Gap & Supply Chain Risk Multiplier

The Governance Gap: MCP servers bypass standard IT vendor procurement. They are installed by individuals but hold organizational keys.



⚠ The **Supply Chain Risk**: With 60+ PyPI releases, a malicious PR pushes code to every developer running `uv tool upgrade`. **One compromised laptop compromises the entire org.**

Permissions Requested vs. Actual Functional Need

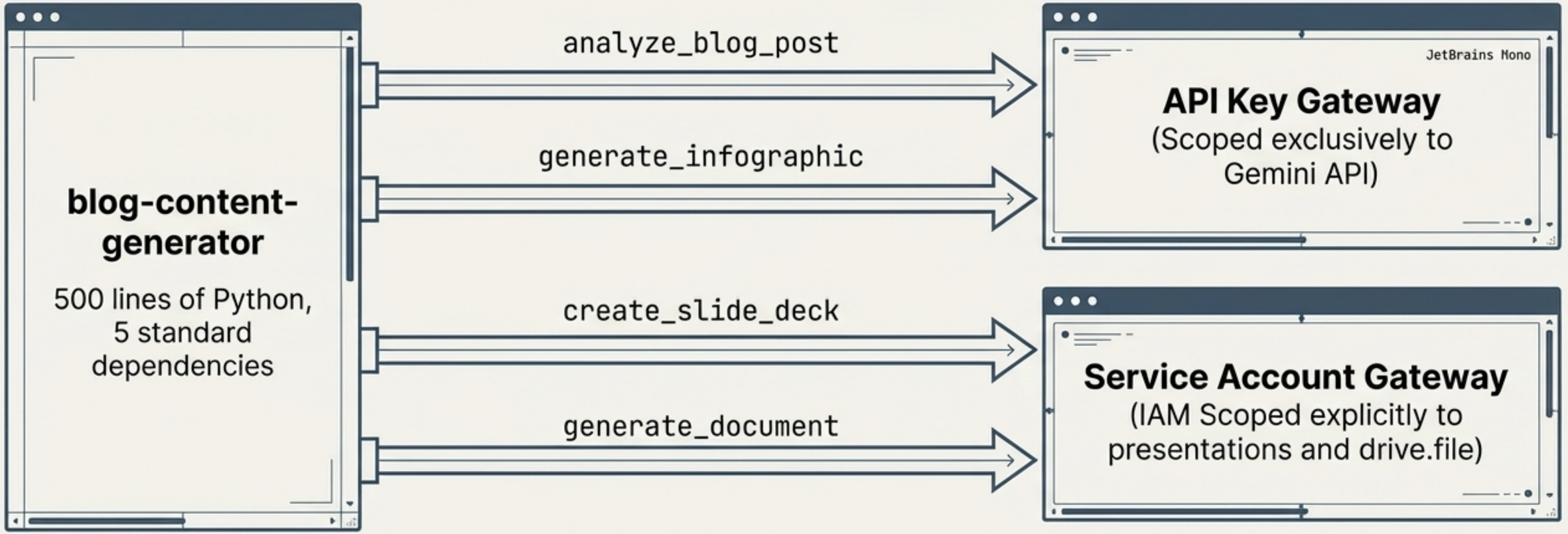


If a tool needs more access than its function requires, that is a design smell.
When permissions far exceed functional requirements: Build, Don’t Borrow.

Technical Specification Comparison: Community vs. Custom Build

Dimension	The Community Tool (`notebooklm-mcp-cli`)	The Custom Build (`blog-content-generator`)
Authentication Scope	Entire Google Identity (Browser Cookies)	Highly Scoped (API Key + Service Account)
Credential Storage	Plaintext JSON (`0o600`)	Environment Variables & Secure Key Files
Attack Surface	Broad (29 tools + CDP exposure)	Minimal (4 validated tools)
API Compliance	Violates ToS (Reverse-engineered RPCs)	Fully Compliant (Official Google SDKs)
Revocation Model	Impossible without killing all sessions	Instantly revoke key/account
Organizational Risk	Enterprise-wide Workspace compromise	Limited to single Drive folder

Editorial AppSec Audit Report



Takeaway: Four tools. Two credentials. Zero browser cookies. Zero ToS violations.

SCENARIO A: THE COOKIE LEAK

```
{
  "cookies": "stolen-session-token",
  "user": "compromised@google.com"
}
```

KILL ALL ACTIVE GOOGLE SESSIONS GLOBALLY.

You cannot revoke just the MCP session; you must log out of your entire digital life.

SCENARIO B: THE KEY/SA LEAK

```
API-KEY: "sk-example-key-123", [redacted]"
SA: "service-account@project.iam.gserviceaccount.com"
```

REVOKE KEY IN CLOUD CONSOLE / DELETE SERVICE ACCOUNT.

Instantly mitigates the threat while leaving personal and enterprise workspaces untouched.

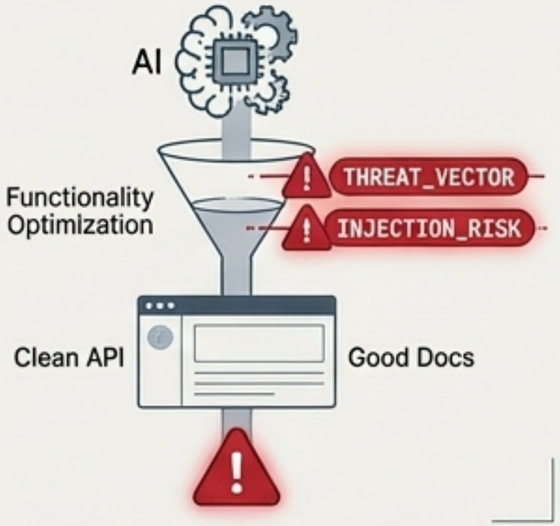
Editorial AppSec Audit Report: State of the Ecosystem

The Wild West

There is no MCP server registry with security ratings. No standardized permission model. The ecosystem is where `npm` was in 2015: move fast, install everything, hope nothing breaks.

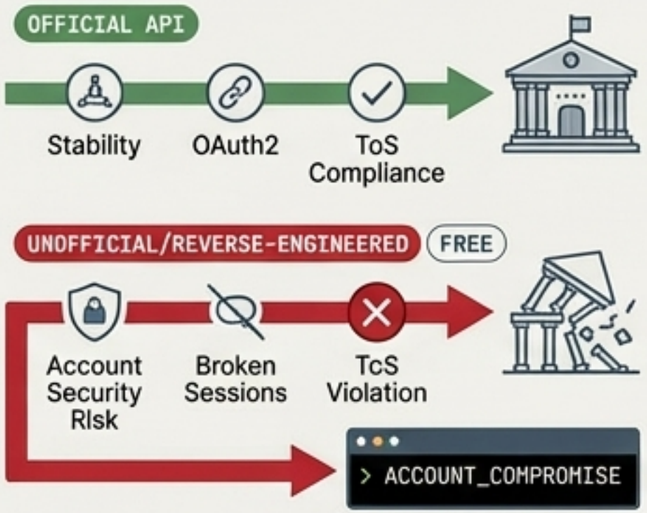
AI Blindspots

Your AI assistant optimizes for functionality, not security. The discoverability pipeline inherently obscures threat vectors behind clean APIs and good documentation.

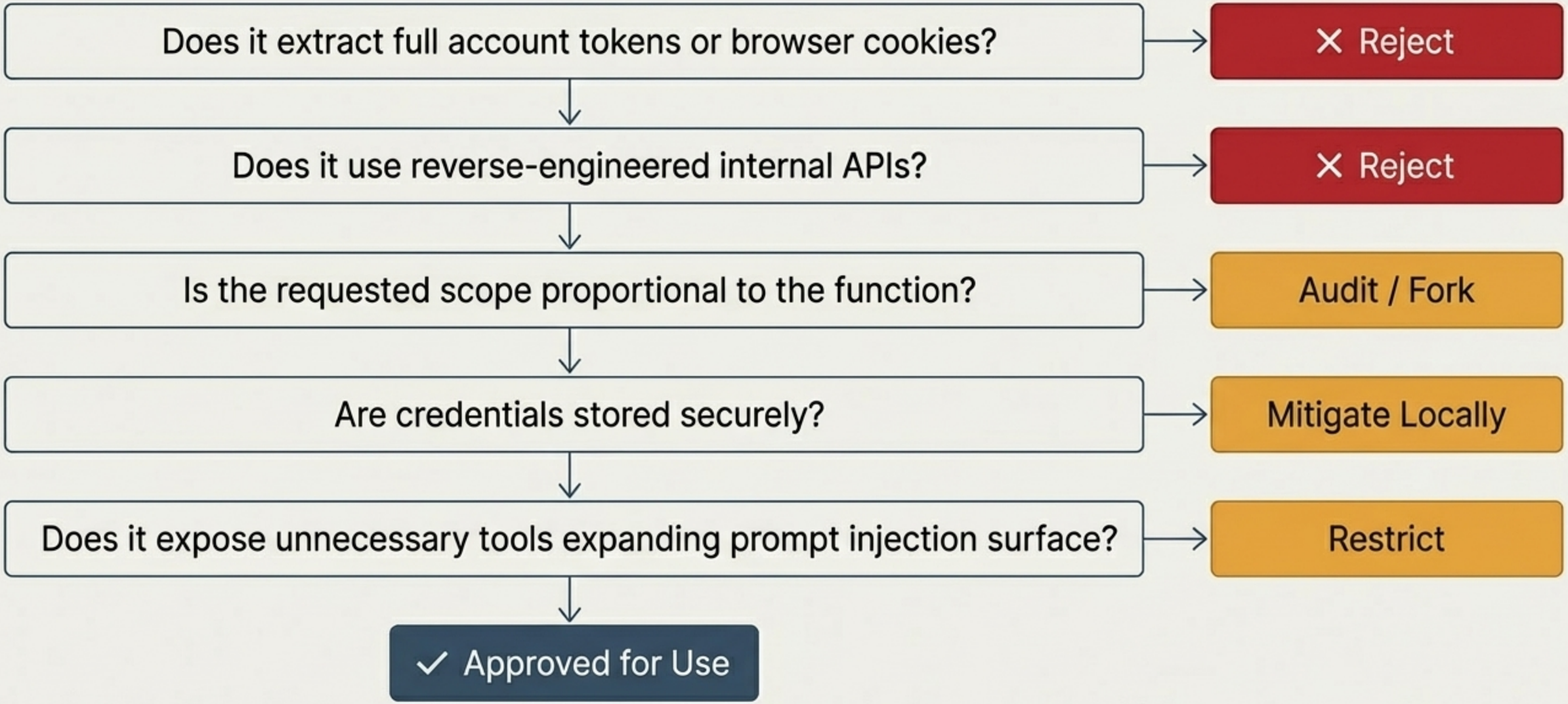


Official > Reverse-Engineered

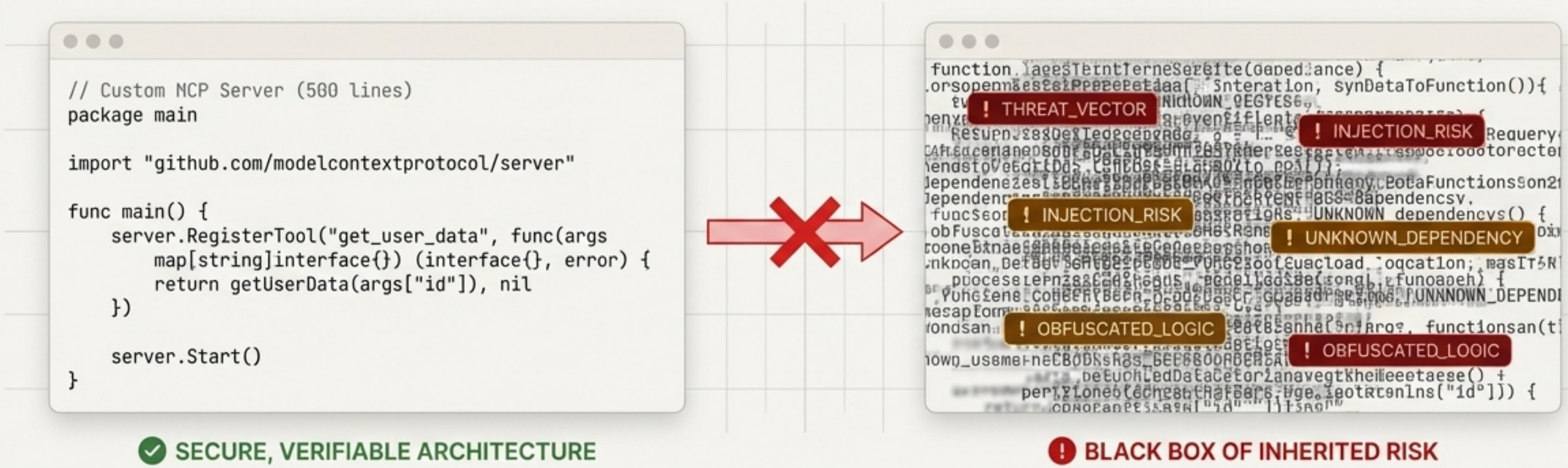
Official APIs cost money but provide stability guarantees, standard OAuth2, and ToS compliance. Unofficial approaches are free but cost you your account security.



MCP Evaluation Scorecard



Build Small. Audit Often.



SECURE, VERIFIABLE ARCHITECTURE

BLACK BOX OF INHERITED RISK

A 500-line MCP server is something you can read, understand, and verify in an afternoon.
A 29-tool server with Chrome DevTools integration is a black box of inherited risk.

The next time an AI assistant suggests a tool that seems perfect, take twenty minutes to check its credentials. Building your own takes less time than mitigating someone else's structural flaws.