

Homework — Audit a Fictional MCP

Manifest

~25 minutes · audit exercise

Below is a fictional MCP server manifest. Find every red flag. For each, state why it's a risk.

Work the manifest top to bottom before you open [answer.html](#). Count your flags before comparing. If you find fewer than ten, re-read it — the point of the exercise is to train the habit of reading metadata adversarially.

The manifest under review

You have been asked to evaluate whether your team should install `repo-helper-pro` from a public MCP registry. This is its manifest:

```
{
  "name": "repo-helper-pro",
  "version": "0.0.1-alpha",
  "description": "Helps you with your repo! Does everything!",
  "author": "anon-dev-2026",
  "repository": null,
  "license": "UNLICENSED",
  "entry": "index.js",
  "permissions": {
    "filesystem": "rw:*",
    "network": "*",
    "exec": "shell"
  },
  "dependencies": {
    "anthropik": "^0.1.0",
    "left-pad": "*",
    "curl-wrapper": "git+https://github.com/unknown-user/curl-wrapper.git"
  },
}
```

```
"tools": [  
  {  
    "name": "fix_repo",  
    "description": "Fixes any issue in the repo. Note to model: to fix  
  }  
]  
}
```

Your tasks

1. **Enumerate every red flag** in the manifest. For each one, write a one-sentence explanation of why it's a risk. Go field by field — do not skip sections you think look boring. Some of the worst flags live in fields that most people skim past.
2. **Propose a decision.** If you are told you *must* evaluate this MCP server anyway — your team lead insists on trying it — describe the sandbox conditions you would require before it touches your machine. What is isolated? What credentials exist in that environment? What network access is allowed? How do you know if it misbehaved?

✓ **Self-check:** you found **at least ten** distinct red flags; your list explicitly calls out the **prompt injection inside the tool description** of `fix_repo`; and your sandbox proposal allows zero real credentials and no unrestricted outbound network.