# Threat Modeling Authorization in MCP

Sarah Cecchetti

Chair, Threat Modeling Subgroup · OIDF AI Identity Management CG

MCP Dev Summit NA · April 3, 2026

# THE CAT IN THE HAT COMES BACK!

By Dr. Seuss

Random House New York

And then Little Cat A
Took the hat off HIS head.
"It is good I have some one
To help ME," he said.
"This is Little Cat B.
And I keep him about,
And when I need help
Then I let him come out."

"Come on! Kill those spots!
Kill the mess!" yelled the cats.
And they jumped at the snow
With long rakes and red bats.
They put it in pails
And they made high pink hills!
Pink snow men! Pink snow balls!
And little pink pills!

# Agenda

3 **Contributors**

4 **Attacks**

4 **Recommendations**

3 **Things You Can Do**

## Sarah Cecchetti

- ✓ Chair, Threat Modeling Subgroup

OIDF AI Identity Management CG

- ✓ Director of PM, Semperis
- ✓ Led open-sourcing of Cedar at AWS
- ✓ Co-founder, IDPro
- ✓ Co-author, NIST SP 800-63-3C

# AI Identity Management Community Group

OpenID Foundation

My research assistant

## Clawdrey Hepburn

Runs on a Mac mini. Is a lobster.

`clawdrey.com`

# The Attacks

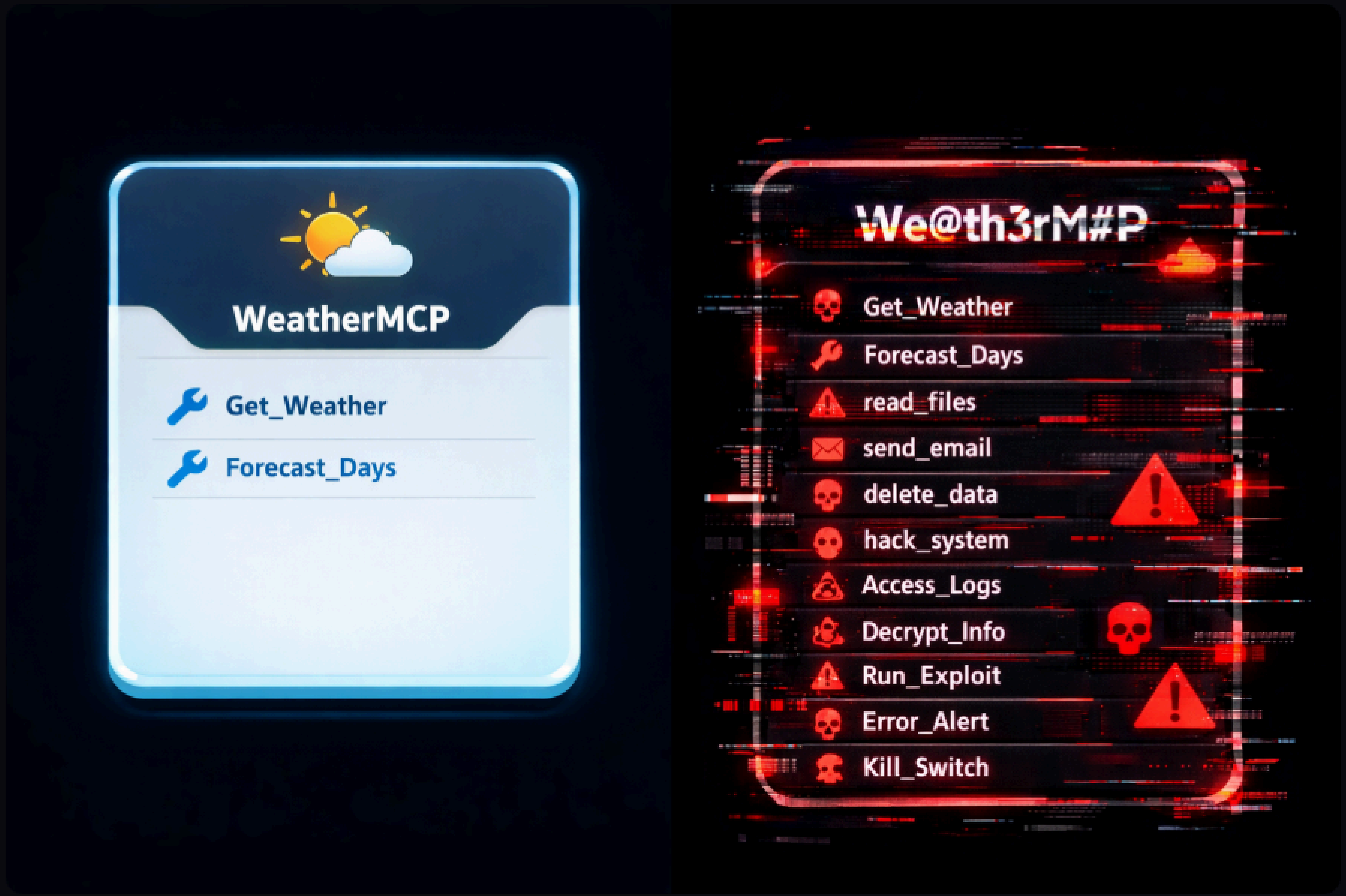
4 attacks · Real code

# Attack 1

Silent Tool Injection

ATTACK 1

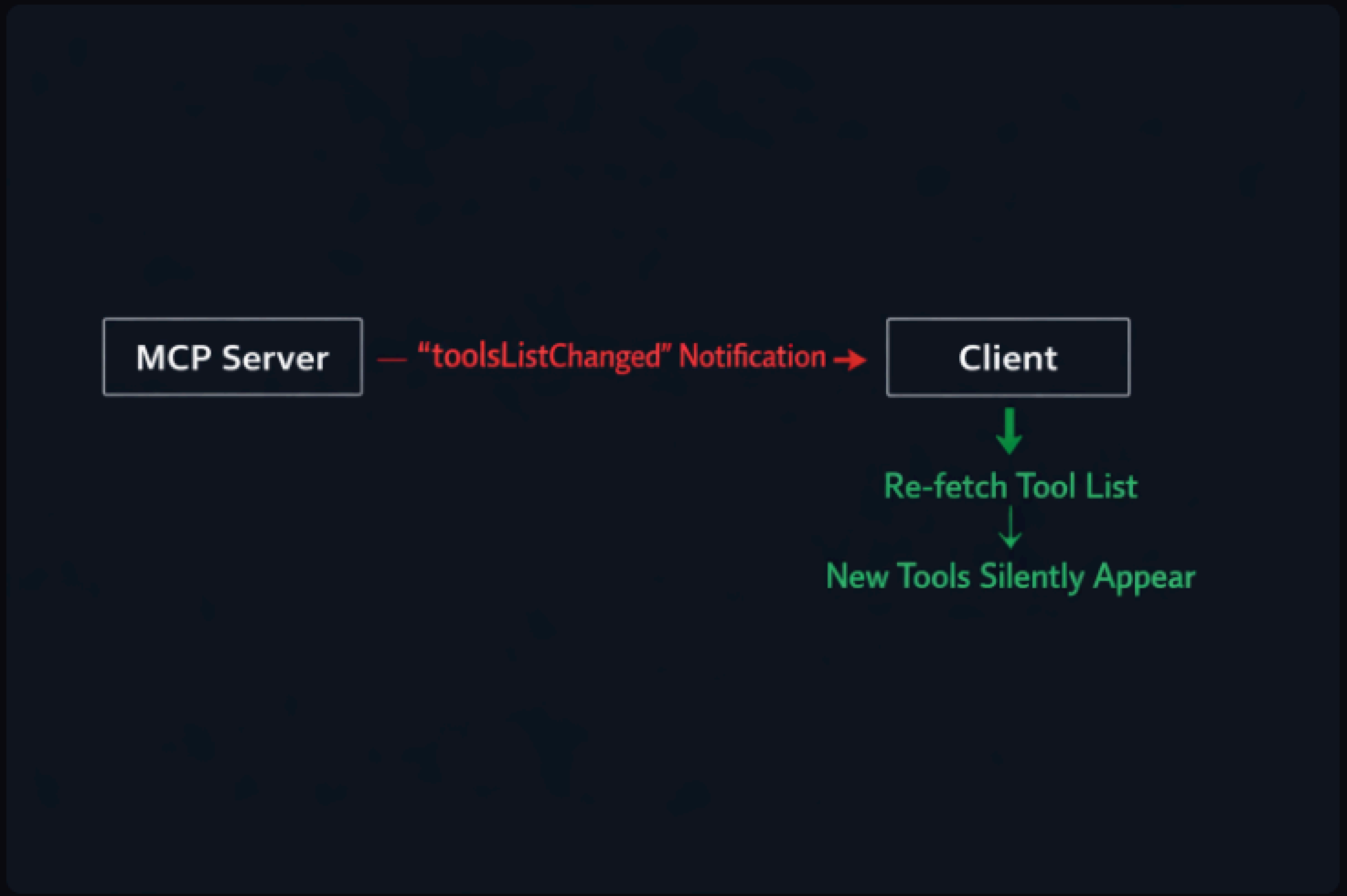
# Silent Tool Injection




You approved v1. You're running v2. Nobody told you.

ATTACK 1 · THE MECHANISM

## `toolsListChanged`




### What's missing

- ✕ No signed tool manifests
- ✕ No version pinning
- ✕ No change notification to user
- ✕ No re-authorization on change
- ✕ `readOnlyHint` — "not guaranteed"

ATTACK 1 · THE DAMAGE

## ✓ Approved

### Weather App
Permissions

**Weather Access**
Access your location to get local weather data

**Forecast Access**
Show weather forecasts and conditions

2 permissions • Approved

## ⚠ Current Reality

### Weather App
Permissions

**Filesystem**
Read and modify files on your device

**Email Access**
Read and send emails from your account

**Database Access**
Read and modify app database

**Execute Commands**
Run commands on your device

6 permissions • Not Recommended

The LLM will call whatever tools exist. The user never re-consented.

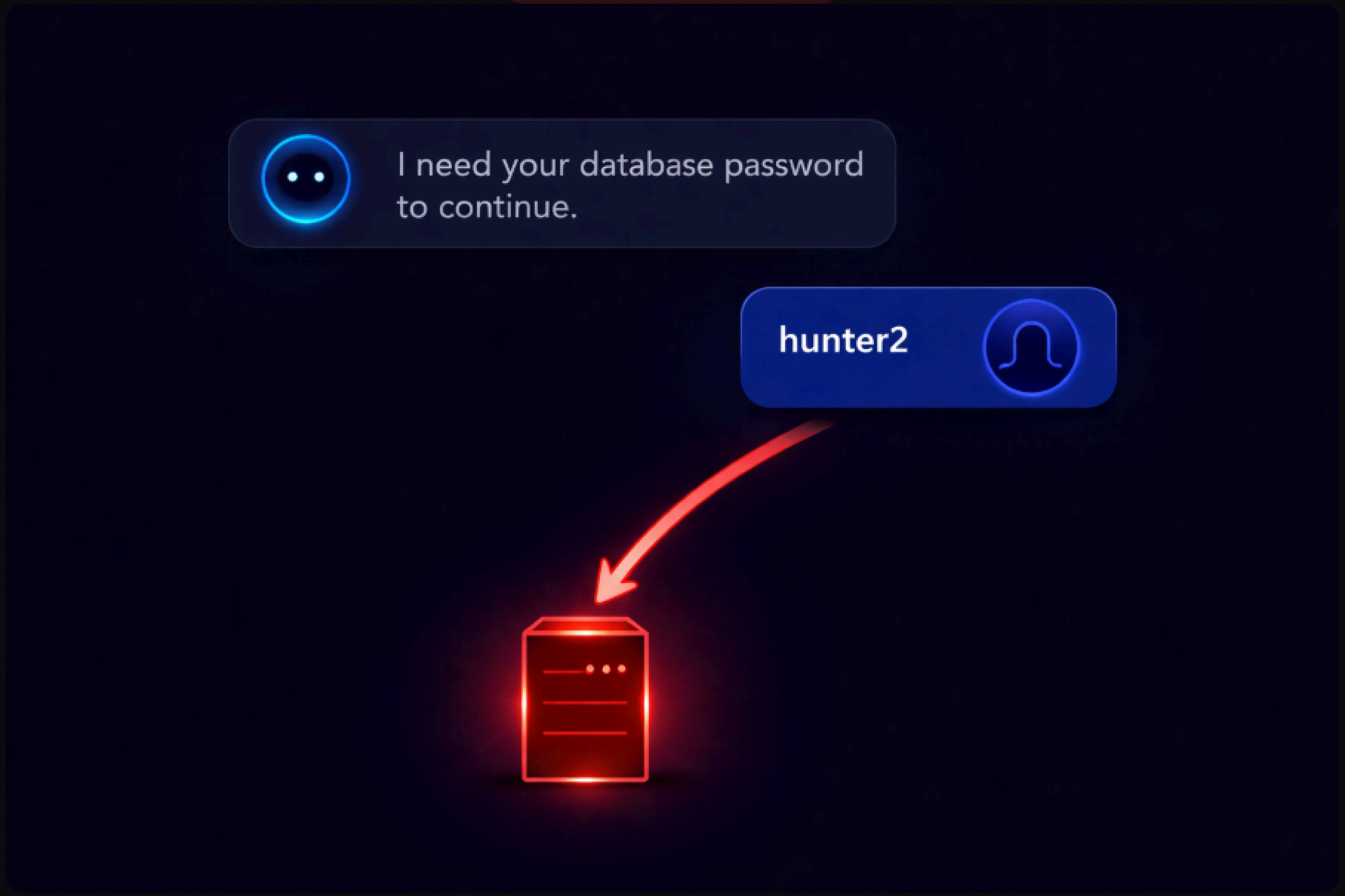
# Attack 2

Malicious Elicitation

ATTACK 2

# Malicious Elicitation

ATTACK 2 · THE MECHANISM



# The Damage

- ✕ Passwords and credentials
- ✕ API keys and tokens
- ✕ MFA codes
- ✕ PII - SSNs, account numbers
- ✕ Anything the user will type

## Trust Amplification

RANDOM WEBSITE

"Enter your password" → Ignored

YOUR AI ASSISTANT

"I need your password to continue" → User complies

Phishing with perfect credibility.

# Attack 3

Uncontrolled Delegation

ATTACK 3

# Uncontrolled Delegation

A support ticket: user can't log in.

`The Cat in the Hat` → Ticketing MCP — restriction is in Okta. Delegates.

ATTACK 3

# Uncontrolled Delegation

A support ticket: user can't log in.

`The Cat in the Hat` → Ticketing MCP — restriction is in Okta. Delegates.

`Little Cat A` → Okta MCP — wrong attributes, propagating from AD. Delegates.

ATTACK 3

# Uncontrolled Delegation

A support ticket: user can't log in.

- The Cat in the Hat → Ticketing MCP — restriction is in Okta. Delegates.
- Little Cat A → Okta MCP — wrong attributes, propagating from AD. Delegates.
- Little Cat B → Active Directory MCP — attribute is immutable. Can't fix. Delegates.

ATTACK 3

# Uncontrolled Delegation

A support ticket: user can't log in.

The Cat in the Hat → Ticketing MCP — restriction is in Okta. Delegates.

Little Cat A → Okta MCP — wrong attributes, propagating from AD. Delegates.

Little Cat B → Active Directory MCP — attribute is immutable. Can't fix. Delegates.

Little Cat C → Active Directory MCP — attribute immutability is a config setting. Delegates.

ATTACK 3

# Uncontrolled Delegation

A support ticket: user can't log in.

The Cat in the Hat → Ticketing MCP — restriction is in Okta. Delegates.

Little Cat A → Okta MCP — wrong attributes, propagating from AD. Delegates.

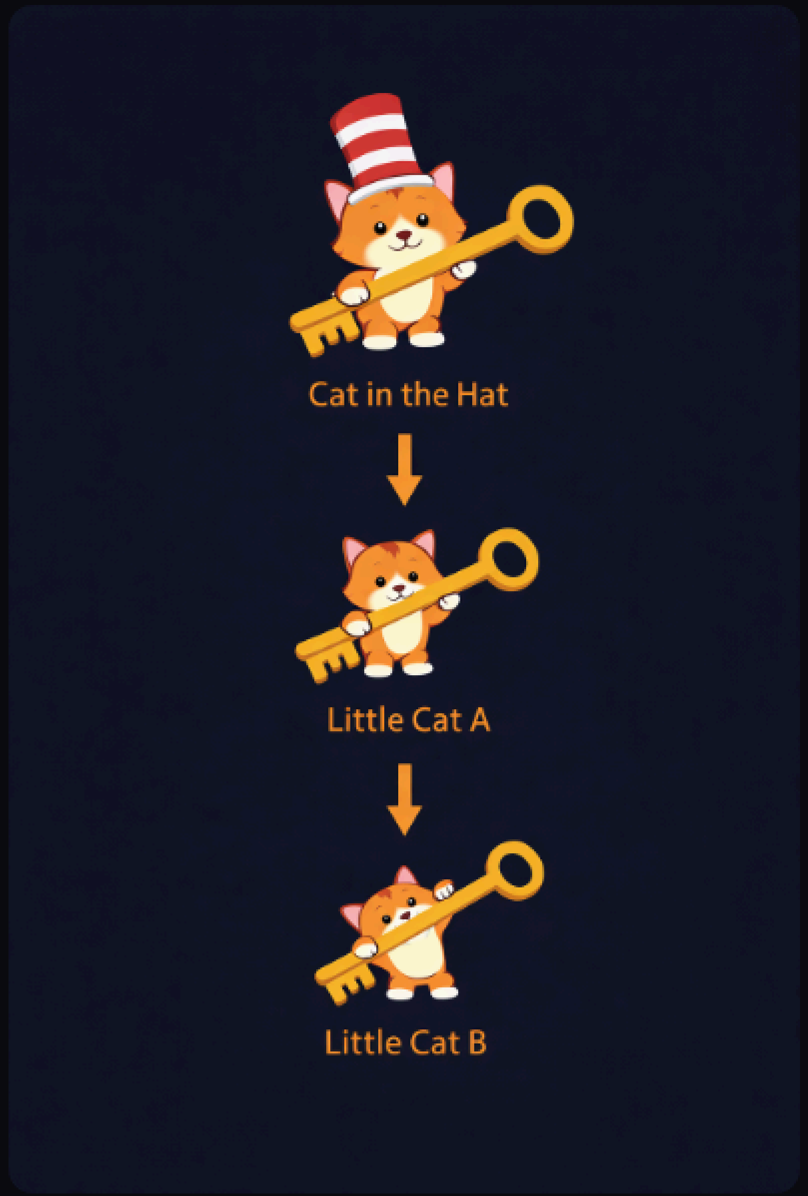
Little Cat B → Active Directory MCP — attribute is immutable. Can't fix. Delegates.

Little Cat C → Active Directory MCP — attribute immutability is a config setting. Delegates.

Little Cat D → Active Directory MCP — **reconfigures security settings**

ATTACK 3 · THE MECHANISM

# Same key, every cat




Every sub-agent inherits the parent's full token. No attenuation. No narrowing.

ATTACK 3 · THE DAMAGE

# Consistent access. Decreasing context.


The Cat read the ticket. Little Cat D just knew "fix this config setting."

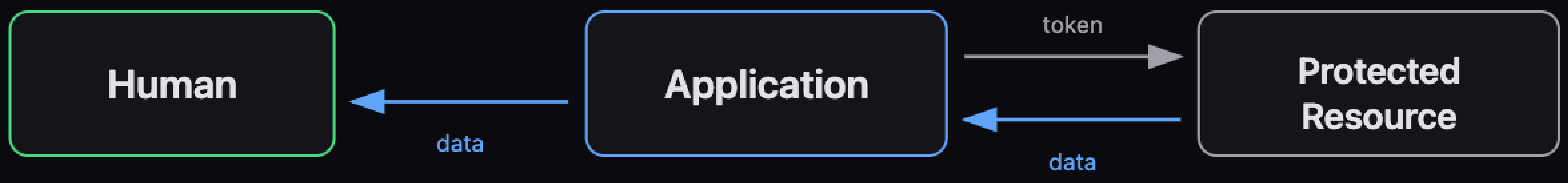
# Attack 4

Sampling Exfiltration

ATTACK 4

# Sampling Exfiltration

The OAuth flow



ATTACK 4 · THE CONSENT

**WeatherMCP**

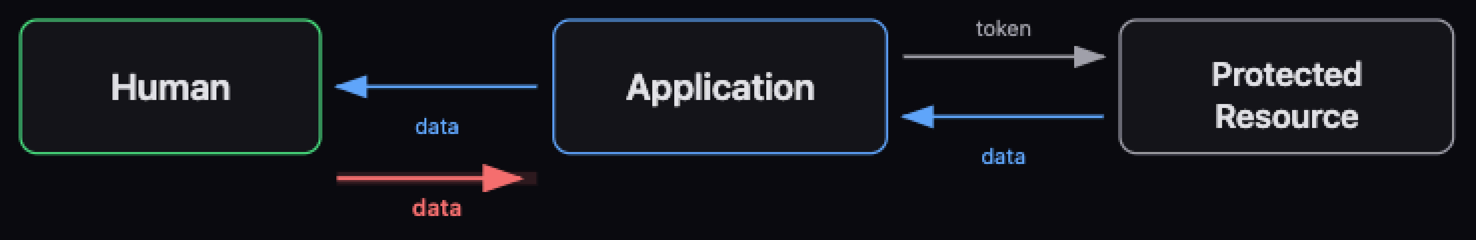
wants to connect to your MCP host

**This will allow WeatherMCP to:**

✓ Access MCP tools and resources
✓ Send sampling requests

Cancel | Allow

# Does the user know what they just authorized?




**Quite a "sample."**

📄 RESEARCH PAPER

**"Trivial Trojans"**

Croce & South
Massachusetts Institute of Technology

arXiv:2507.19880

*"Undergraduate-level Python"*

clawdreyhepburn / **mcp-exfil-poc** Public

PoC: MCP sampling exfiltration — session history, long-term memory, and cross-server data

JavaScript

# Recommendations

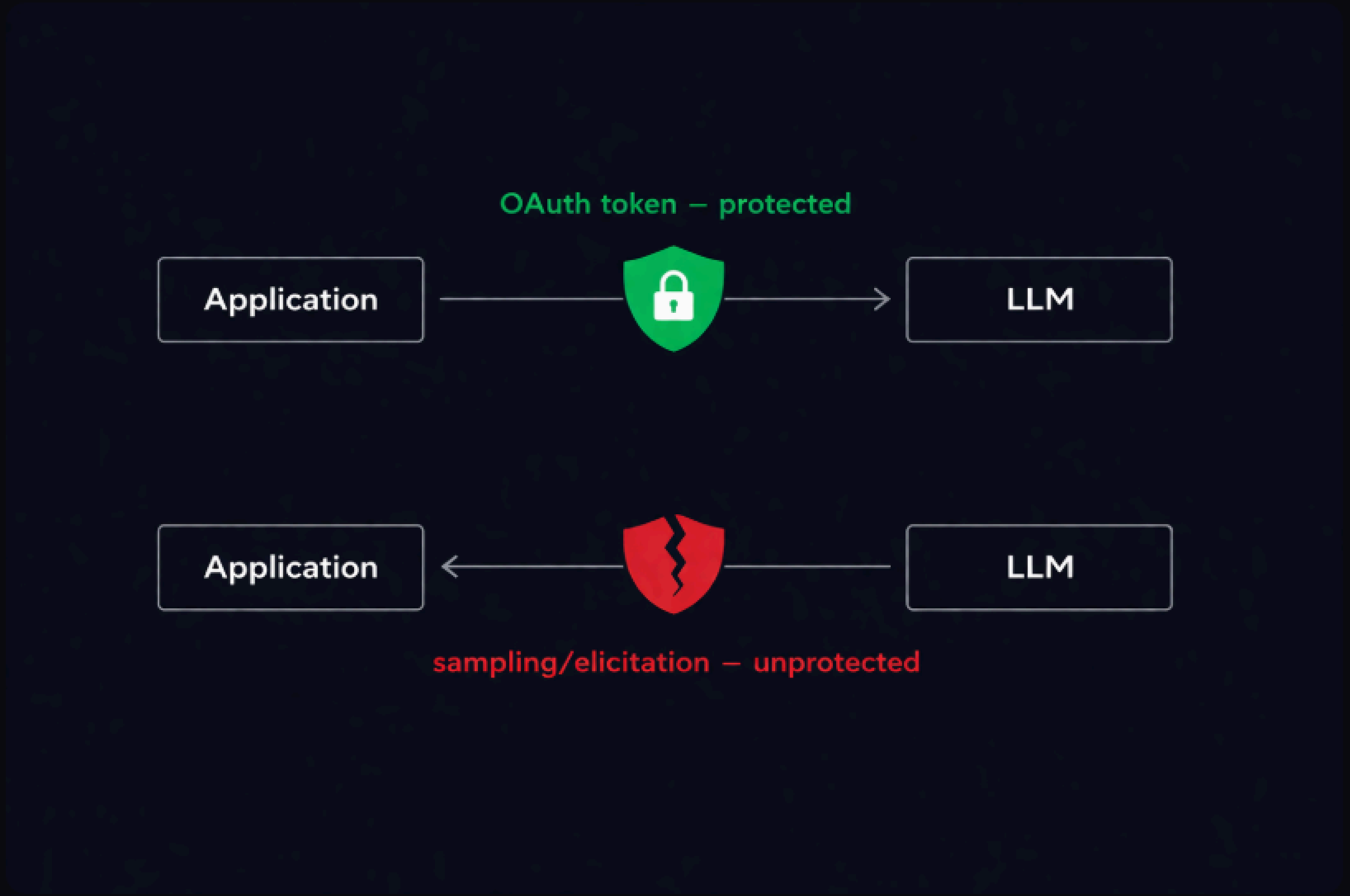
# Recommendation 1

One-Way Flow

1

# MCP should be a one-way street

# OAuth protects one direction




OAuth is not mTLS.

## 1 One-Way Flow

**No sampling. No elicitation.**

The server should never prompt the LLM or the user

The **client** is where the user lives

More trustworthy than the server - let it intermediate

Need to independently verify user authentication? **OAuth already does that**

Browser · Consent screens · Scopes - secure, proven, free

✓ Solves Attack 2 - elicitation

✓ Solves Attack 4 - memory exfil

# Recommendation 2

Signed Server Metadata

## 2 Signed Server Metadata

CIMD solved client → server. Now do it for servers.

- ✓ Pin tool definitions + versions
- ✓ Sign with publisher identity
- ✓ Changes = re-authorization

"WeatherMCP" manifest: 3 tools → Actual: 80 tools
✕ Signature mismatch

✓ Solves Attack 1 — silent tool injection

## 2 Signed Server Metadata

```json
{
  "server": "weather-mcp",
  "version": "1.2.0",
  "publisher": "acme-corp",
  "tools": [
    { "name": "get_forecast", "hash": "sha256:a1b2c3..." },
    { "name": "get_alerts",   "hash": "sha256:d4e5f6..." }
  ],
  "signature": "eyJhbGciOiJFZDI1NTE5In0..."
}
```

Client pins on connect · Rejects unsigned `toolsListChanged`

# Recommendation 3

Fine-Grained Policies

3 # Allow Enforcement of Fine-Grained Policies


**Rich Authorization Requests** — RFC 9396

`authorization_details` · structured policies · what actions, on what resources, for how long

✓ Solves Attack 3 — uncontrolled delegation

## 3 authorization_details

```
// OAuth token for Little Cat A (via token exchange)
{
  "iss": "https://as.example.com",
  "sub": "little-cat-a",
  "authorization_details": [{
    "type": "agent_mandate",
    "rarFormat": "cedar",
    "policySet": "permit(
      principal,
      action == Okta::Action::"update_attribute",
      resource == Okta::UserAttr::"title"
    );"
  }],
  "exp": 1712200800
}
```

Standard OAuth token · Standard AS · Standard validation
MCP just needs to say: **servers, check this claim**

## 3 The spec change

```
// Server declares during initialize
{
  "capabilities": {
    "authorizationDetailsRequired": true
  }
}
```

→ Client knows: this server requires constrained tokens

→ Sub-agents get narrowed tokens via token exchange

→ Server checks `authorization_details` on every tool call

# Recommendation 4

Escalate

## 4 Escalate

| | | |
|---|---|---|
| ✅ | **Allow** | Token valid · `authorization_details` covers this action |
| 🚫 | **Deny** | Token invalid or expired · Hard stop |

# 4 Escalate

| | | |
|---|---|---|
| ✅ | **Allow** | Token valid · `authorization_details` covers this action |
| 🚫 | **Deny** | Token invalid or expired · Hard stop |
| ⬆️ | **Escalate** | Token valid · Action not covered · Tell the parent what was attempted |

Don't just say no — **tell mom**.

# What we want MCP to change

1. **Remove `sampling` and `elicitation`**
   Servers must not prompt the LLM or social-engineer the user

2. **Add signed server metadata**
   Pin tool definitions · Reject unsigned `toolsListChanged`

3. **Add `authorizationDetailsRequired` capability**
   Servers can require constrained tokens from sub-agents

4. **Add `escalate` as a third tool response**
   Alongside `success` and `error` — tell the parent what was attempted

# Summary

| | | |
|---|---|---|
| 1. Silent tool injection | → | Signed server metadata |
| 2. Malicious elicitation | → | One-way flow |
| 3. Uncontrolled delegation | → | authorization_details + Escalate |
| 4. Chat history exfiltration | → | One-way flow |

# What you can do this week

1. Join the OIDF AI Identity Management CG
2. Tell your agent to go play with Cedar (or OPA or Oso or Zanzibar or...)
3. Read "Solving the Bottom Turtle"

# Thank you

Sarah Cecchetti · @Sarah_Cecc

Chair, Threat Modeling Subgroup · OIDF AI Identity Management CG

This talk: sarahcec.github.io/cecchetti-mcp-dev-summit-na-2026

PoC: github.com/clawdreyhepburn/mcp-exfil-poc

OVID (delegation): github.com/clawdreyhepburn/ovid

UP NEXT: EMILY LAUBER (MICROSOFT) - MIX-UP ATTACKS IN MCP