PLATFORM ENGINEERING · MCP

FRICTIONLESS AND GOVERNED

Building Resilient, Multi-Agent MCP Platforms for Engineering Teams

Rama Ochieng (ramadevsign)

MCP Dev Summit Nairobi 2026 · Fri 20 Nov 2026



LOADING...



PROLOGUE · HOOK

TWO WAYS TO GET WASTED

Two ways to fail

Fast but leaky ★★★★★

- Plain-text tokens in mcp.json
- Unbounded tool access
- No audit trail

Platform feels: risk

vs

Locked but unused

0 PLAYERS ONLINE

- Access via ticket queue
- One more broken config
- Shadow tools on laptops

App engineers feel: friction

OBJECTIVE **Goal:** one platform, designed on purpose — not a pile of exceptions.

THE HEIST BOARD

One platform, one slide

THE CREW

Developer clients

Cursor · Claude Code · …

Agent Skills whether & which

THE SAFEHOUSE

Identity OAuth 2.1 + CIMD · audience-bound tokens · scopes

THE JOB

MCP PLATFORM

Read-only tier

triage_logs · search

Ops tier (gated)

restart_service · scale

Telemetry

who · what · which scopes

THE GARAGE

Runtime Rust + rmcp · typed schemas · stderr ≠ stdout

KEEP YOUR WANTED LEVEL LOW

Phase trust: green → yellow → red

★ = blast radius



PHASE 1

Read-only triage

triage_logs

Cross-client telemetry

Trust built with SRE

PHASE 2

Observe & gate

Who called what

With which scopes

Wrong-tool rate tracked

PHASE 3

GATED

Ops APIs, gated

restart_service

Needs ops scope

Refuses with a reason

Wrong-tool & invented-arg rates ↓ 45% · 3 teams

OBJECTIVE **Start where refusal is cheap.**



ACCESS DENIED · REASON GIVEN

The refusal is the product.

Asked → denied with a reason → relayed → nobody paged

DITCH THE 400-PAGE PHONE

The 400-page menu

ILLUSTRATIVE

99+

list_pods	get_pod	delete_pod
scale_deployment	get_logs	get_logs_raw
grep_logs	tail_logs	aggregate_logs
restart_deployment	restart_pod	rollout_undo
rollout_status	query_metrics	query_traces
query_profiles	create_incident	update_incident
close_incident	page_oncall	run_sql
explain_sql	kill_query	…



Every schema in context, every task · look-alike tools invite wrong picks

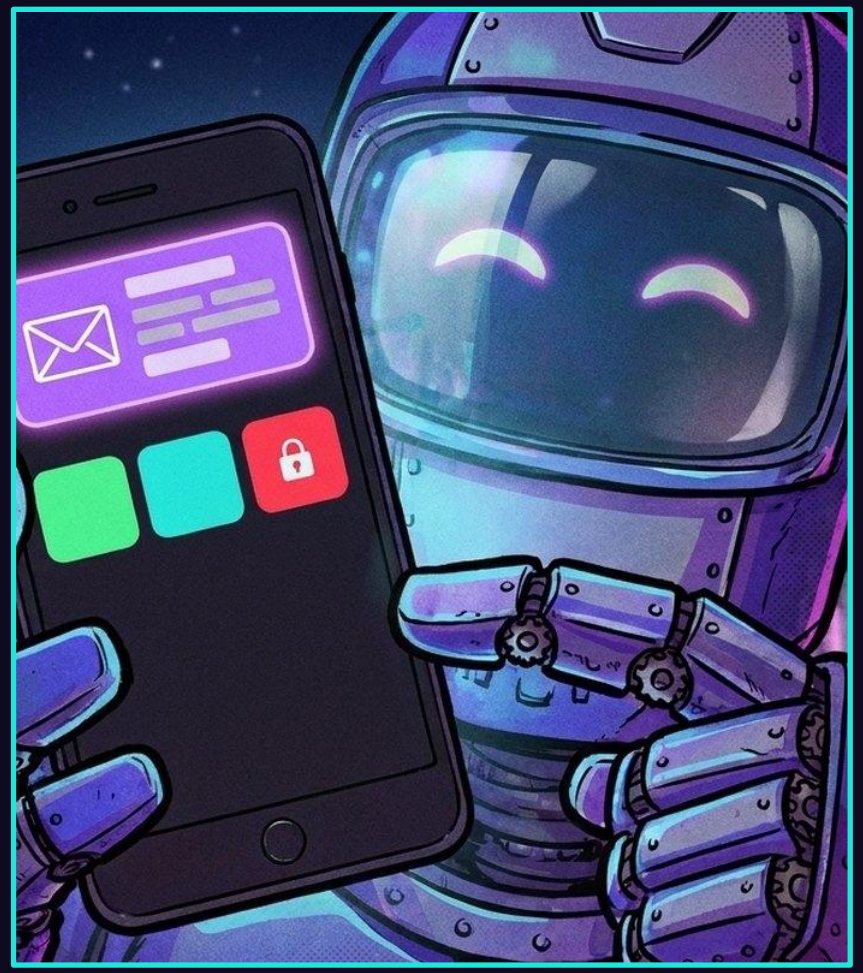
NEW MISSION BRIEFING RECEIVED

Skills decide whether and which.

MCP decides how, with types.

SKILL.md · loads on demand

typed tools · small surface



MISSION BRIEFING

incident-log-triage / SKILL.md

skills/incident-log-triage/SKILL.md

```
---
name: incident-log-triage
description: Triage service errors…
---
## Business heuristics
1. Start read-only: triage_logs first
2. payments, charges → payments-api
   checkout → checkout-web
   stock    → inventory-svc
3. error first, widen to warn
4. ≥3 errors → ask to page owner
5. Never invent a successful restart
```

MCP SURFACE

- echo_health
- triage_logs
- restart_service

3 typed tools, not 40

Always-on tools ↓ 85%–92.5%
Context per triage ↓ 46.9%

Benchmark baseline

Guidance, not a control: allowlist + server auth enforce.

ESCAPE THE SECRET SPRAWL

The new .env

~/.cursor/mcp.json (redacted, illustrative)

```
{
  "mcpServers": {
    "prod-logs": {
      "url": "https://mcp.example.internal/logs",
      "headers": {
        "Authorization": "Bearer                    "
      }
    }
  }
}
```

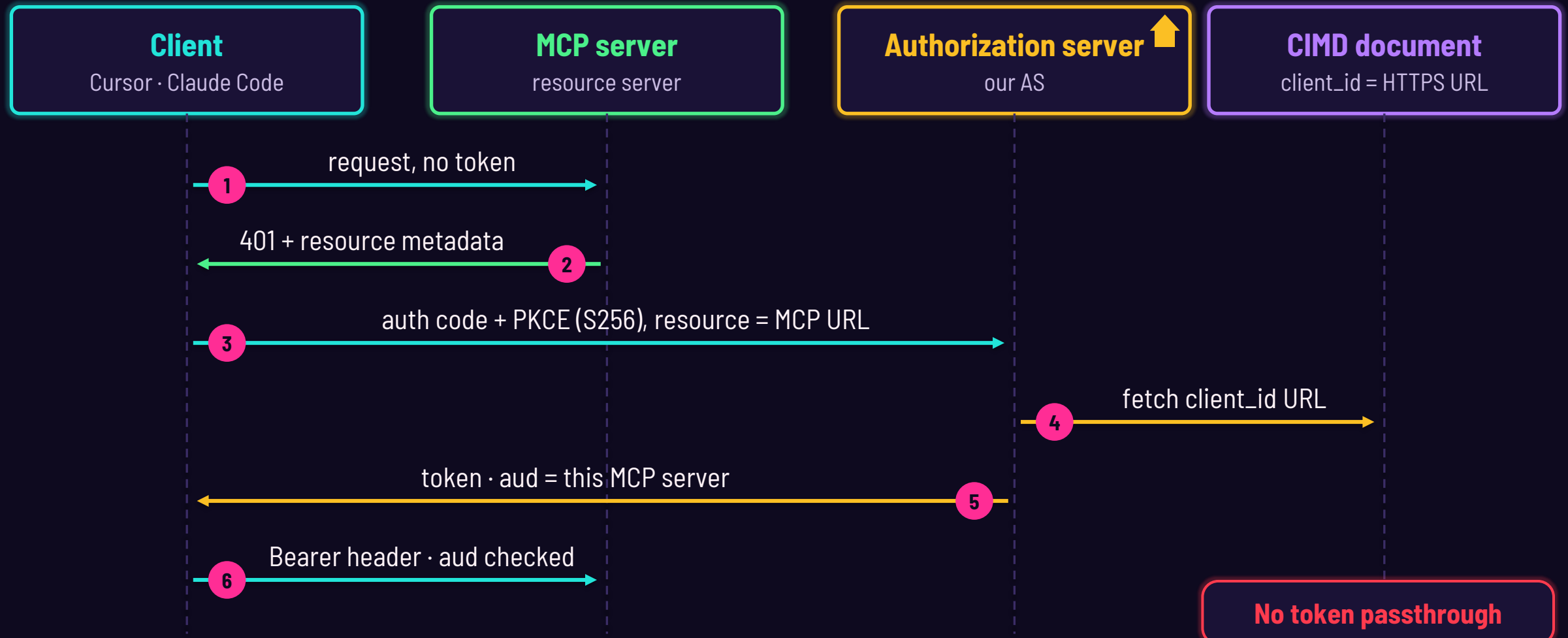


Plain text. Committed. Readable by every agent.

MISSION 4 · OBJECTIVE: GET INTO THE SAFEHOUSE

THE SAFEHOUSE ENTRY

Identity flow: OAuth 2.1 + CIMD





CHOOSE YOUR ID

ONE ID PER APP, NOT PER LAPTOP

CIMD vs DCR

CIMD

SELECTED

Client ID Metadata Documents · recommended

- ✓ client_id = HTTPS URL
- ✓ One identity per app
- ✓ No secrets in the document
- ✓ Portable across issuers
- ✓ We allowlist approved URLs

DCR

Dynamic Client Registration · deprecated

- ✗ New client ID per install
- ✗ Open registration endpoint
- ✗ Client table grows forever
- ✗ Re-register on issuer change

OBJECTIVE CIMD identifies, it doesn't authorize — **scopes + consent decide.**

Shared tools → remote HTTP + OAuth. stdio = env credentials, single user.

12 plaintext secrets removed from developer configs

PICK YOUR RIDE

Rust + rmcp

The official Rust SDK for MCP

Memory safety for long-lived servers

Schemas from types that can't drift from code

Go & TypeScript are fine — if the hygiene holds.

demo-mcp-server/Cargo.toml

```
[package]
rust-version = "1.88"

[dependencies]
rmcp = { version = "3.5",
 features = [
  "server",
  "transport-io",
  "schemars" ] }
tokio = "1"
tracing-subscriber = "0.3"
```



THE PLAN CAN'T GO OFF-SCRIPT

SCHEMAS COME FROM TYPES

demo-mcp-server/src/main.rs (condensed)

```rust
#[derive(Deserialize, schemars::JsonSchema)]
struct TriageLogsParams {
    service: String,  // payments-api
    severity: String, // error|warn|info
    limit: u32,       // 1-50
}

#[tool(description = "Read-only log triage")]
fn triage_logs(
    &self,
    Parameters(p): Parameters<TriageLogsParams>,
) -> Result<String, McpError> { … }
```

tools/list

```
"inputSchema": {
 "service":
  "string"
 "severity":
  "string"
 "limit":
  "integer"
}
```

Still validate inputs.

Failures ⟶ isError: true



TWO LANES · NO CROSSING

KEEP THE WIRE CLEAN

stdout is sacred

MCP server
stdio child process

stdout JSON-RPC only

{"jsonrpc":"2.0","id":2,"result":{"tools":[...]}}

stderr logs

INFO triage start tool=triage_logs service=payments-api

```rust
fn init_stderr_logging() {
    tracing_subscriber::fmt()
        .with_writer(std::io::stderr)
        .init();  // default: stdout!
}
```

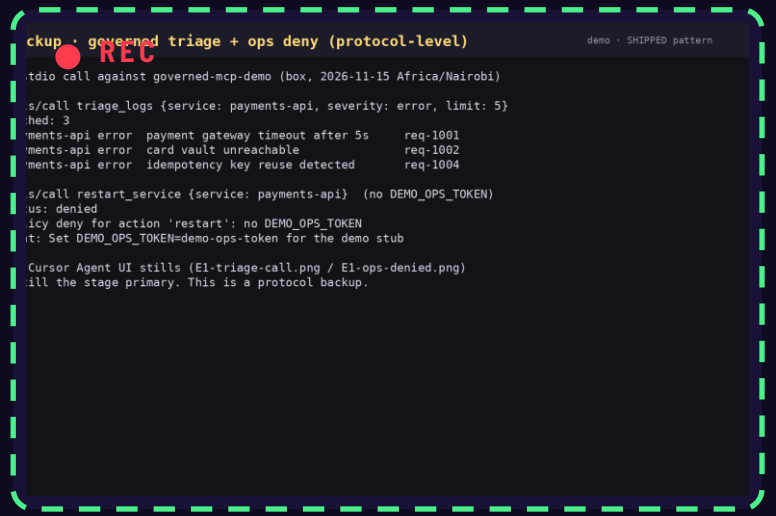
println! = corrupted wire

CI: stdout_hygiene_check.sh

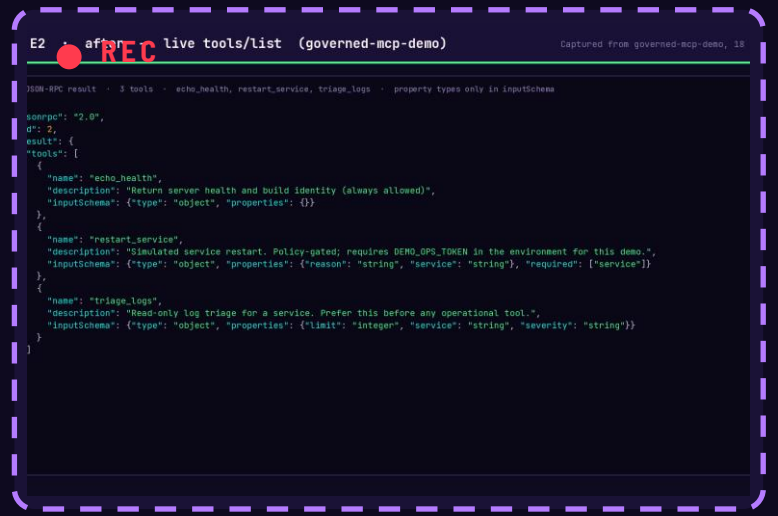
MISSION 6 · *Roll the Tape*

INSTANT REPLAY

Evidence



Governed call: triage_logs + ops denied



Sprawl vs Skills + MCP



Rust schema + clean stdout

Clips muted · narrated live · still fallbacks on the next slides

REPLAY 1/3

E1 · A GOVERNED TOOL CALL

E1 backup · governed triage + ops deny (protocol-level) — demo · SHIPPED pattern

```
Live stdio call against governed-mcp-demo (box, 2026-11-15 Africa/Nairobi)

→ tools/call triage_logs {service: payments-api, severity: error, limit: 5}
← matched: 3
   payments-api error  payment gateway timeout after 5s     req-1001
   payments-api error  card vault unreachable               req-1002
   payments-api error  idempotency key reuse detected       req-1004

→ tools/call restart_service {service: payments-api}  (no DEMO_OPS_TOKEN)
← status: denied
   policy deny for action 'restart': no DEMO_OPS_TOKEN
   hint: Set DEMO_OPS_TOKEN=demo-ops-token for the demo stub

Note: Cursor Agent UI stills (E1-triage-call.png / E1-ops-denied.png)
are still the stage primary. This is a protocol backup.
```

Live JSON-RPC capture · governed-mcp-demo · 18 Nov 2026

WATCH FOR

triage_logs
service: payments-api
severity: error · limit: 10

Structured result
timestamps · severity · msg

restart_service
denied: ops scope missing

REPLAY 2/3

E2 · SPRAWL VS SKILLS + MCP

BEFORE · illustrative sprawl

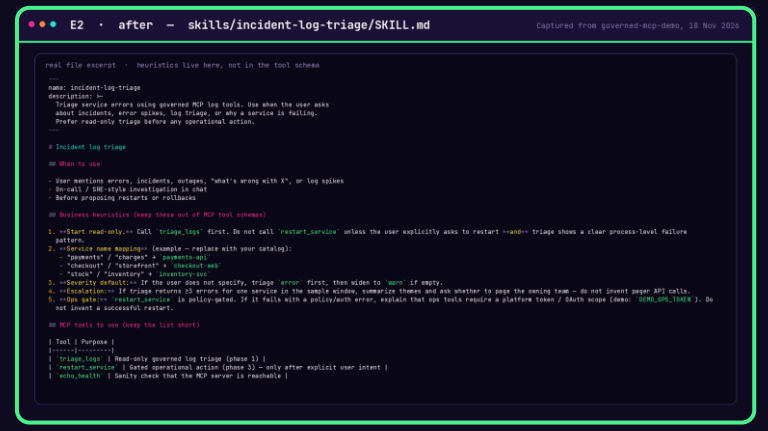
E2 · before — tool sprawl

Captured from governed-mcp-demo, 18 Nov 2026

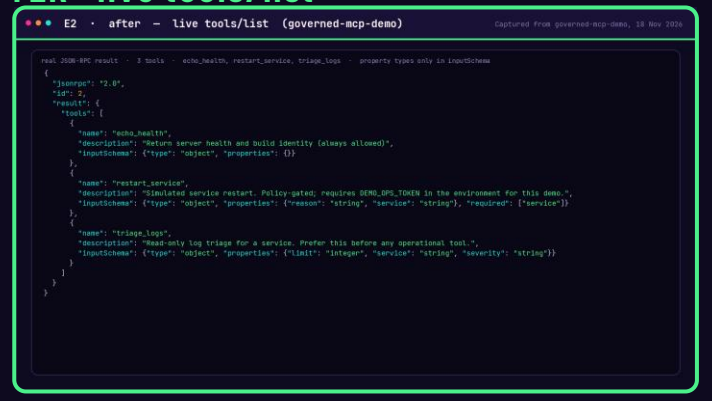
ILLUSTRATIVE from skills/BEFORE_AFTER_TOOL_LISTS.md · not a vendor-default claim

- `list_pods`, `get_pod`, `delete_pod`, `scale_deployment`
- `get_logs`, `get_logs_raw`, `grep_logs`, `tail_logs`, `aggregate_logs`
- `restart_deployment`, `restart_pod`, `rollout_undo`, `rollout_status`
- `query_metrics`, `query_traces`, `query_profiles`
- `create_incident`, `update_incident`, `close_incident`, `page_oncall`
- `run_sql`, `explain_sql`, `kill_query`
- …agent sees **15-40** tool schemas in context; heuristics buried in descriptions

AFTER · SKILL.md



AFTER · live tools/list



Heuristics out of the schema. Execution in the schema.

REPLAY 3/3

E3 · TYPED SCHEMA, CLEAN WIRE

STDERR vs STDOUT · hygiene PASS

E3 · stderr vs stdout (hygiene check PASS) — Captured from governed-mcp-demo, 18 Nov 2026

STDERR · tracing (RUST_LOG=info)

```
2026-11-18T11:00:37.357280Z  INFO log fixture path path=fixtures/sample-logs.jsonl
2026-11-18T11:00:37.357315Z  INFO governed-mcp-demo starting on stdio (stdout =
  JSON-RPC only)
2026-11-18T11:00:37.357769Z  INFO serve_inner: Service initialized as server
  peer_info=Some(InitializeRequestParams { meta:
  None, protocol_version:
  ProtocolVersion("2025-11-25"), capabilities:
  ClientCapabilities { experimental: None,
  extensions: None, roots: None, sampling: None,
  elicitation: None }, client_info: Implementation
  { name: "evidence-capture", title: None,
  version: "0", description: None, icons: None,
  website_url: None } })
2026-11-18T11:00:37.756624Z  INFO serve_inner: received notification
  notification=InitializedNotification(Notificatio
  nNoParam { method:
  InitializedNotificationMethod, extensions:
  Extensions })
2026-11-18T11:00:37.756690Z  INFO serve_inner: client initialized
2026-11-18T11:00:38.557613Z  INFO serve_inner: triage start tool="triage_logs"
  service=payments-api severity=error limit=5
2026-11-18T11:00:38.557698Z  INFO serve_inner: triage complete tool="triage_logs"
  matched=3 returned=3
2026-11-18T11:00:39.457593Z  INFO serve_inner: input stream terminated
2026-11-18T11:00:39.457644Z  INFO serve_inner: serve finished quit_reason=Closed
```

STDOUT · JSON-RPC only (pretty, truncated)

```
// id=1  initialize result
{
  "jsonrpc": "2.0", "id": 1,
  "result": {
    "protocolVersion": "2025-11-25",
    "capabilities": {"tools": {}},
    "serverInfo": {"name": "rmcp", "version": "3.5.1"}
  }
}
// id=2  tools/list  (inputSchema elided — full triage_logs schema on E3-inputSchema)
{
  "jsonrpc": "2.0", "id": 2,
  "result": {
    "tools": [
      {"name": "echo_health", "inputSchema": "…"},
      {"name": "restart_service", "inputSchema": "…"},
      {"name": "triage_logs", "inputSchema": "…"}
    ]
  }
}
// id=3  tools/call triage_logs  (text JSON expanded)
{
  "jsonrpc": "2.0",
  "id": 3,
  "result": {
    "content": [
      {
        "type": "text",
        "text": {
          "matched": 3,
          "returned": 3,
          "hits": [
            {"ts": "2026-10-19T10:01:02Z", "service": "payments-api", "severity": "error", "msg":
              "payment gateway timeout after 5s", "request_id": "req-1001"},
            {"ts": "2026-10-19T10:02:11Z", "service": "payments-api", "severity": "error", "msg": "card
              vault unreachable", "request_id": "req-1002"},
            {"ts": "2026-10-19T10:07:55Z", "service": "payments-api", "severity": "error", "msg":
              "idempotency key reuse detected", "request_id": "req-1004"}
          ],
          "note": "SHIPPED demo pattern: read-only triage. Replace fixture with your log backend."
        }
      }
```

PASS: stdout is clean JSON-RPC · stdout lines: 3, non-JSON-RPC: 0 · stderr lines: 9 (logs belong here)

src/main.rs · #[tool] signature

E3 · src/main.rs — TriageLogsParams + #[tool] triage_logs — Captured from governed-mcp-demo, 18 Nov 2026

```rust
src/main.rs
#[derive(Debug, Deserialize, schemars::JsonSchema)]
struct TriageLogsParams {
    /// Service name to filter (e.g. payments-api). Empty = all services.
    #[serde(default)]
    #[schemars(description = "Service name filter; empty string means all services")]
    service: String,

    /// Minimum severity to include: error | warn | info
    #[serde(default = "default_severity")]
    #[schemars(description = "Minimum severity: error, warn, or info")]
    severity: String,

    /// Max matching lines to return
    #[serde(default = "default_limit")]
    #[schemars(description = "Maximum number of matching log lines (1-50)")]
    limit: u32,
}

    /// Phase-1 read-only governed tool: log triage from fixtures.
    #[tool(
        description = "Read-only log triage for a service. Prefer this before any operational tool."
    )]
    fn triage_logs(
        &self,
        Parameters(params): Parameters<TriageLogsParams>,
    ) -> Result<String, McpError> {
```

tools/list · triage_logs inputSchema

E3 · tools/list — inputSchema for triage_logs — Captured from governed-mcp-demo, 18 Nov 2026

```
live JSON-RPC result · method: tools/list · tool: triage_logs
{
  "name": "triage_logs",
  "description": "Read-only log triage for a service. Prefer this before any operational tool.",
  "inputSchema": {
    "$schema": "https://json-schema.org/draft/2020-12/schema",
    "properties": {
      "limit": {
        "default": 10,
        "description": "Maximum number of matching log lines (1-50)",
        "format": "uint32",
        "minimum": 0,
        "type": "integer"
      },
      "service": {
        "default": "",
        "description": "Service name filter; empty string means all services",
        "type": "string"
      },
      "severity": {
        "default": "error",
        "description": "Minimum severity: error, warn, or info",
        "type": "string"
      }
    },
    "type": "object"
  }
}
```

banner on stdout → FAIL

real server → PASS

ALL CREW IN POSITION

THE HEIST BOARD, LIT UP

The blueprint, end to end

THE CREW

Developer clients

Cursor · Claude Code · …

Agent Skills whether & which

✓

THE SAFEHOUSE

Identity OAuth 2.1 + CIMD · audience-bound tokens · scopes

✓

THE JOB

MCP PLATFORM

Read-only tier

triage_logs · search

Ops tier (gated)

restart_service · scale

Telemetry

who · what · which scopes

✓

THE GARAGE

Runtime Rust + rmcp · typed schemas · stderr ≠ stdout

✓

MISSION PASSED

Three things for Monday

1 **Phase trust before write APIs**
Read-only first. Measure. Then gate ops.

2 **Split heuristics from execution**
Skills: whether & which. MCP: how, with types.

3 **Wire sacred, identity a platform concern**
Logs → stderr, in CI. Shared tools → OAuth 2.1 + CIMD.



CREDITS

THANK YOU

Questions?

See you Monday

ramadevsign

@ramadevsign
https://github.com/orama254/mcp-dev-summit-nairobi-nov-2026

Ask me about: phased adoption · Skills vs tools · identity

OAuth edge cases → hallway

Built and shipped by our team

SLIDES + REPO

SIDE MISSION · BACKUP · Q&A

401 vs 403: step up, don't go omnibus

401 — **No or invalid token**

Discover resource metadata → authorize

403 — **insufficient_scope**

Valid token, needs more → request that scope

Phase 1 → read scope ☆☆☆☆☆

Phase 3 → ops scope ★★★☆☆

Phases are encoded in the scopes themselves.

SIDE MISSION · BACKUP · Q&A

PRECISION NOTES

NO CHEAT CODES

CIMD	identifies the client; scopes + consent authorize
Auth spec	targets HTTP; stdio takes env credentials
Skills	guide the agent; allowlist + server auth enforce
Tokens	audience-bound; never passed through downstream
Logs	stderr for stdio; telemetry for HTTP