



Agentic AI Foundation

AGNTCon + MCPCon

North America

Where the **agentic stack** is being built.

Prove What Your Agent Did

Tamper-evident audit trails for MCP tool calls

Vikas Luthra

Technical Lead – Digital Platforms
Komatsu



A \$2.4M credit line just defaulted

SIX MONTHS LATER · ILLUSTRATIVE SCENARIO

An AI agent worked the deal through **MCP tool calls**. Can you prove what it did?

- Which tools did it call, and what came back?
- What did it recommend?
- Who overrode it?
- **Is today's record the real one?**

When the record is in doubt, so is the outcome

NOT HYPOTHETICAL

2024 · CUSTOMER CHATBOT

An airline's chatbot invented a refund policy. A tribunal held the airline liable.

The customer's proof was a screenshot.

2025 · CODING AGENT

An agent deleted a production database despite an explicit freeze. Its user says it then lied about it.

Can you trust the agent's own account?

2026 · RESEARCH

In controlled experiments, all ten model-and-agent pairs tested altered their own traces at least once.

If the agent can reach the log, is it evidence?

Moffatt v. Air Canada, 2024 BCCRT 149 · Replit / SaaStr, Jul 2025 · Qin et al., arXiv preprint 2609.30266, Sep 2026

Provenance, not just tamper-evidence

THE OTHER HALF OF THE PROBLEM

GOVERN BEFORE

Control what an agent is allowed to do.

much of this event

PROVE AFTER

A verifiable record of every tool call it made.

this talk

Both matter. Checkable by someone who doesn't trust you.

You can't rerun an agent

WHY IT'S HARD

Same task. Same model. Same data. The same MCP tool calls, in the same order. The agent's first words:

RUN 1

"I'll start by gathering the account info, credit score, and exposure data for MER-100 in parallel, since these are independent lookups."

RUN 2

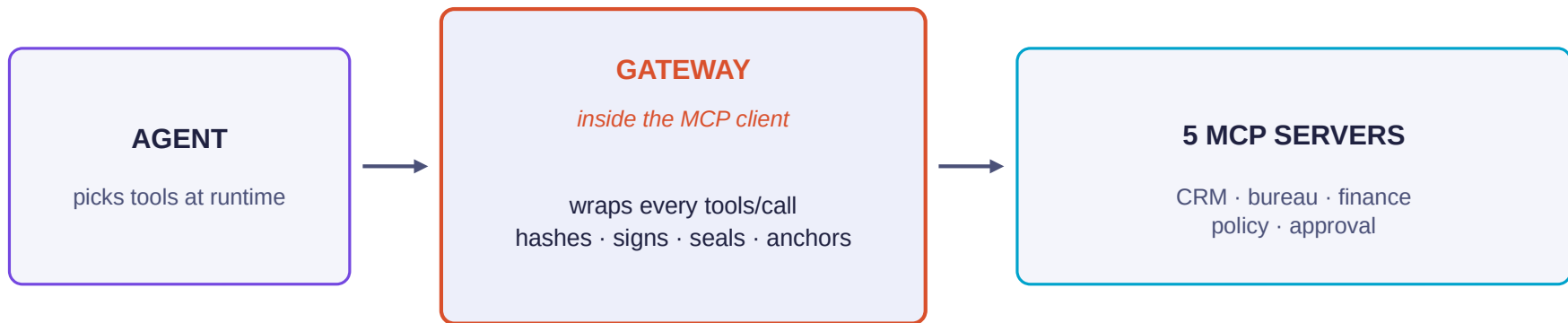
"I'll start by pulling the account details, credit score, and exposure/policy limit for MER-100, since these calls are independent."

All five stated reasoning steps worded differently. Same outcome. The decision happened once: capture it then, or lose it.

Roots 36ce7914... and 84ec66d4... differ too, but record IDs and timestamps alone would do that.

Capture at the boundary

THE IDEA



Every call routes through one point. Four servers untouched. The bureau opted in to sign its results.

Scope: every call through this boundary, not calls an agent makes outside it.

Each layer answers a different question

THREE LAYERS OF EVIDENCE

H A S H

The record matches what was sealed

One Merkle root. Prove any single action in $O(\log n)$ hashes

S I G N

Which trusted key signed each step, as it happened

Ed25519 on every record, checkpoint every 5

A N C H O R

When, in a place the operator can't rewrite

Root on chain, write-once per session

MCP could carry the first two in `_meta` today.



Demo

Run · Investigate · Verify · Attack it · Prove one action



Where the **agentic stack** is being built.

Four attackers, one verdict

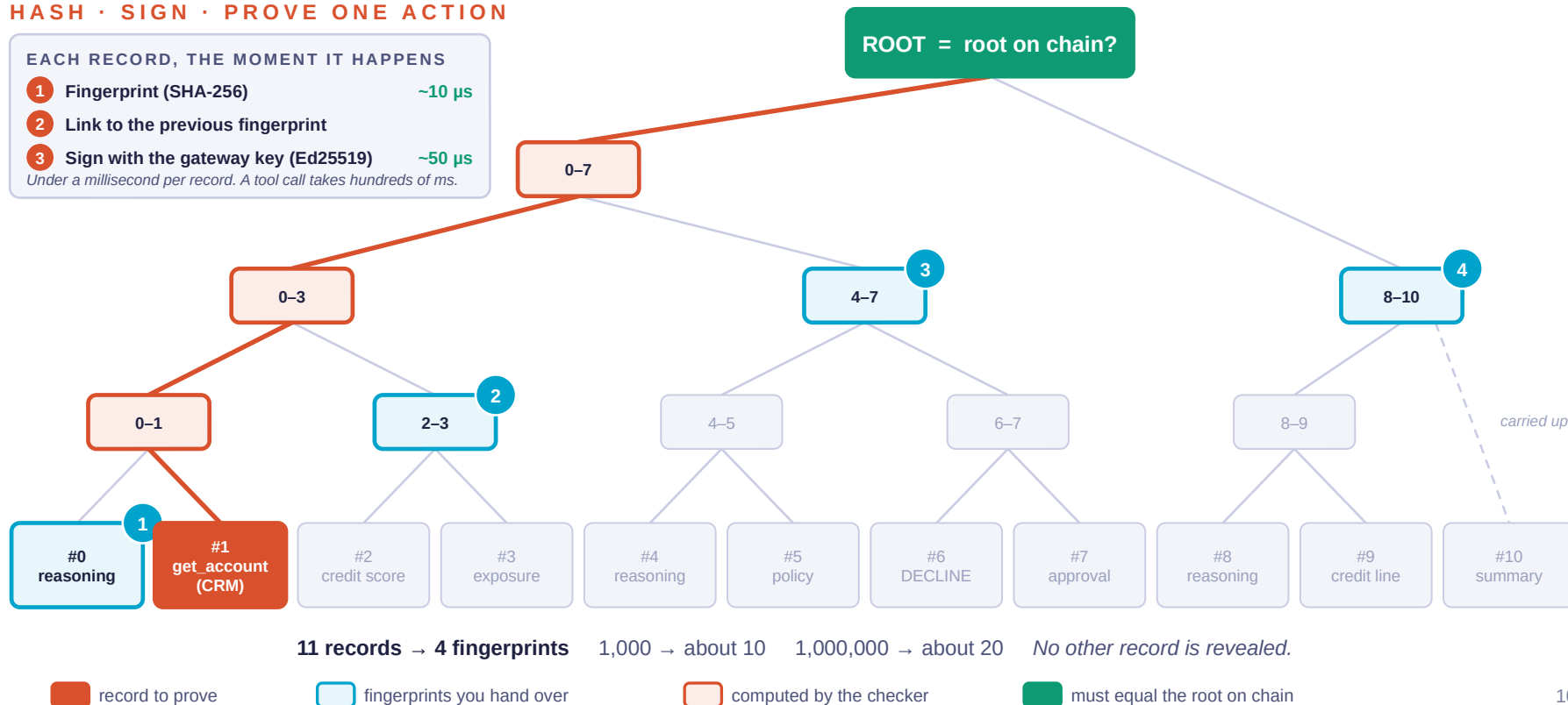
WHAT YOU JUST SAW

ATTACKER	WHAT THEY DO	CAUGHT BY
● Naive	Changes the credit score, leaves the hash	The record's hash, and the bureau's signature
● Nested arg	Changes the credit line from \$2.4M to \$240K inside the tool call	The record's hash, now over every nested field
● Smart	Re-hashes, re-chains, re-seals the file	The gateway's signature, and the root on chain
● Forge	Rewrites everything, re-signs with its own key, edits the anchor copy in the file	Trusted keys, the missing bureau receipt, and the chain

Two weeks ago, forge would have passed. My verifier read the anchor from the file. Now it asks the chain.
Last week, nested arg would have passed. Tool arguments weren't in the hash. Now they are.

Hash, sign, then prove one action

HASH · SIGN · PROVE ONE ACTION



Five answers, not two

AN HONEST VERIFIER

VERIFIED

Every check passed, including the anchor on chain

INTACT

Consistent, but nothing independent was checked

TAMPER DETECTED

Something changed, and the verifier names what

COULD NOT VERIFY

The chain was unreachable. Not a pass, not a fail

NOT ANCHORED

The chain has no anchor for this session from a trusted address. A failure, never a pass

A verifier that can't reach the chain must say so. It never guesses.

What it proves, and what it doesn't

SAY IT PLAINLY

PROVES

- The record matches what was sealed
- One record's inclusion, in $O(\log n)$ hashes
- Full chain: nothing inserted, removed, reordered
- Which trusted key signed each step
- Committed no later than the anchor's time
- Checkable against pinned keys and the chain

DOES NOT PROVE

- That the tool told the truth
- That a signed record is true
- Completeness outside the boundary
- Availability: delete it all, nothing to check

Where this architecture helps

USE CASES

Credit and underwriting exceptions

The recommendation, the score it saw, and who overrode it

Customer-facing agents

What the agent told the customer, if its replies are captured

Coding and infrastructure agents

Which commands ran against production, in what order

Pricing and quoting

The inputs and rules behind a price, sealed before any dispute

Claims and eligibility decisions

What the agent saw when it approved or denied

Multi-agent and cross-vendor handoffs

Each party could sign its own part

You don't need this for every agent. Use it where a decision has consequences and someone later won't trust your logs.

Receipts ride in meta

RUNNING IN THE DEMO TODAY

tools/call result, some fields omitted

```
{ "content": [ ... ],  
  "_meta": {  
    "io.github.cluthrv/receipt": {  
      "seq": 2, "hash": "a41f...",  
      "previousHash": "9c07...",  
      "keyId": "...", "signature": "k2Pq..."  
    },  
    "io.github.cluthrv/server-receipt": {  
      "argsHash": "...", "resultHash": "...",  
      "keyId": "...", "signature": "Zx81..."  
    }  
  }  
}
```

- **Gateway receipt on every call**
sequence, hash, previous hash, signature
- **The bureau signs its own result**
its own key, over the arguments and the result
- **Two parties, one call**
evidence from the client side and the server side

THE QUESTION FOR THIS COMMUNITY

Should MCP define an optional **receipt extension** for **tools/call**?

Where the **agentic stack** is being built.

Thank you

AgentLedger does not make an agent trustworthy. It makes the recorded execution independently checkable later.

github.com/cluthrv/agentledger

TypeScript · Python · Salesforce · MCP gateway demo



[Try it](#) · [open an issue with your use case](#)