MCP SERVER | NO CODE | CLOUD HOSTED

# Incident to Claim Timeline MCP

## Turn messy insurance data into a clear, ordered history.

Incident to Claim Timeline MCP organizes fragmented insurance data into a single, chronological record. Your AI client uses this to merge incident events, witness statements, and policy milestones into a factual sequence. It helps you verify reporting compliance and track the time elapsed since an incident occurred.

A+ Quality Score 100/100

claims | timeline | insurance | compliance | chronology

MAINTAINED BY VINKIUS CLOUD FOR AI AGENTS · SEPTEMBER 2026

vinkius.com/en/ai-agent-connect/incident-to-claim-timeline

SCAN TO VIEW

# The connectivity layer between AI and the world's software.

Vinkius sits between AI and every application. All communication passes through Vinkius Cloud via the Model Context Protocol (MCP) — with governance, observability, and security at every layer.

# Your AI Connections Run Through Vinkius Cloud

The world's largest managed MCP catalog

Vinkius is the connectivity layer where AI connects to the software your business already runs. We handle the hosting, the security, the credentials, the uptime — you get agents that actually do things.

We operate the world's largest managed MCP catalog. Major SaaS platforms, CRMs, databases, and cloud providers — running, monitored, production-ready. This MCP server is hosted and maintained by the Vinkius Cloud for AI Agents.

> *The agent doesn't manage credentials, doesn't manage uptime, doesn't manage security. Vinkius does.*
>
> — Architecture principle

## Four Pillars of the Vinkius Runtime

**01 — Security by design**

Credentials stay encrypted at rest via AES-256. The AI agent never touches raw keys — they're injected into a sandboxed V8 isolate at runtime. Actions are logged, and connections have an emergency kill switch.

**02 — Built on MCP Fusion**

This MCP server was built with **MCP Fusion**, the open-source framework (Apache 2.0) that powers the entire Vinkius catalog. Schema-as-firewall strips undeclared fields, compiled PII redaction runs at zero overhead, and cryptographic lockfiles produce git-diffable audit trails.

**03 — Deterministic observability**

Eight immutable metrics per endpoint: request volume, p95 latency, error rate, active connections, cost attribution. A live payload feed logs every tool call with mutation detection.

**04 — Autonomous operations**

Servers are deployed, monitored, and patched autonomously. New capabilities and security patches ship weekly. Zero-downtime deployments ensure continuous availability across all managed MCP servers.

| AES-256 | Ed25519 | 24h TTL | V8 Isolate |
|---|---|---|---|
| Encryption at rest | PKI vault signatures | Ephemeral session keys | Sandboxed execution |

## One Token. Instant Access.

Every MCP server on Vinkius is accessed through a **Connection Token**. Tokens are generated in the cloud dashboard and produce a unique MCP endpoint URL. Paste this URL into any MCP-compatible client — no SDK required.

A single token can serve **multiple AI clients simultaneously**, or you can issue separate tokens per client for granular access control. Each token tracks its own request count, last activity timestamp, and can be individually enabled or revoked.

MCP ENDPOINT

```
https://edge.vinkius.com/{token}/mcp
```

Claude · Cursor · VS Code · Windsurf · Grok · Gemini

## Security Is the Architecture

Security in Vinkius is not a feature — it's the foundation of the runtime. The gateway enforces multiple independent protection layers between AI agents and third-party APIs.

**01 — Ed25519 PKI Vault**

Every workspace has an Ed25519 Master Key. Session keys are generated ephemerally (24h TTL) and signed by the Master Key. Credentials never leave the vault boundary.

**02 — V8 Isolate Sandboxing**

Tool code runs inside isolated-vm V8 isolates with 64 MB memory caps and per-request timeouts. No filesystem access, no network access except through the SSRF-guarded fetch bridge.

**03 — SSRF Guard**

All outbound HTTP requests are DNS-resolved and validated before execution. Private IP ranges (10.x, 172.16-31.x, 192.168.x, AWS metadata 169.254.x) are blocked at the network layer.

**04 — DLP & PII Redaction**

A ResponseGuard pipeline intercepts every tool response. Configurable redaction patterns strip sensitive fields (emails, SSNs, card numbers) before data reaches the AI agent.

**05 — Cryptographic Audit Trail**

Every request is signed into a SHA-256 hash chain with Ed25519 signatures. Events form a tamper-proof, SIEM-exportable forensic record.

**06 — Honeytoken Trap System**

Phantom credentials are injected into isolated environments. If a honeytoken is used outside Vinkius infrastructure, the server is quarantined instantly.

# Emergency Kill Switch

EU AI Act Art. 14(1) Compliant

The kill switch is an **emergency halt** mechanism — not a simple toggle. When triggered, it executes three actions atomically:

**01 — Server deactivated**

The MCP server is immediately taken offline across the entire cluster.

**02 — All tokens revoked**

Every connection token is invalidated. Total lockout — reconnection blocked until new tokens are issued.

**03 — WebSocket connections killed**

Active connections terminated via Redis pubsub broadcast. Propagates to every runtime node in the cluster.

## Full Visibility. Zero Guesswork.

The Vinkius cloud dashboard includes a full MCP Governance suite — real-time analytics and security controls for production AI operations.

**Control Plane**

KPI dashboard with request volume, latency, success rate, token consumption, and AI-generated operational briefings.

**Agent Activity**

Which AI clients are connecting, how often, and what they're doing — real-time session tracking.

**FinOps**

Cost tracking per tool, payload compression savings, budget optimization signals, and consumption trends.

**Tool Health**

Slowest and most error-prone tools, with actionable root-cause insights and performance baselines.

**Firewall & DLP**

PII redaction activity, sensitive data protection counters, and security event timeline.

**Incident Log**

Error trends, failure rates, status-code breakdowns, and forensic audit trail access.

> Get started at **cloud.vinkius.com** — connect your AI agent in under 60 seconds.

# Incident to Claim Timeline MCP

4 tools available
Cloud-hosted on Vinkius

Managing insurance claims often means digging through a disorganized pile of dates, notices, and statements. This MCP fixes that by reconstructing a complete, ordered history from your raw data. You can feed your agent various data points, and it will build a single timeline that shows exactly how an incident progressed through the policy lifecycle.

Instead of manually checking if a notice was sent on time, your AI client can run compliance checks against policy milestones. You can also isolate specific parts of the history, like just the witness statements or just the policy notices, to get a clearer view of specific phases. It turns a collection of disconnected timestamps into a logical narrative that makes sense for claims adjusting and auditing.

## Core Capabilities

**01 — Timeline Reconstruction**

Your agent builds a full, ordered history from scattered incident data.

**02 — Compliance Checking**

The AI verifies if reporting happened within required policy windows.

**03 — Data Filtering**

Your client isolates specific categories like notices or milestones from the full timeline.

**04 — Status Summarization**

The agent generates a narrative of the time passed since the incident occurred.

## One Click on Vinkius — From Prompt to Execution

Available at **vinkius.com/en/ai-agent-connect/incident-to-claim-timeline** — connect your AI agent in three steps.

01 Connect the MCP to your AI client via Vinkius.

02 Provide your incident data, witness statements, or policy milestones to your agent.

03 Ask your agent to build a timeline or check for compliance.

04 Review the structured, chronological output or narrative summary.

Get your insurance data organized in minutes by connecting this MCP to your preferred AI client.

## Built For

This MCP is built for insurance professionals who need to audit claim histories and verify timelines quickly.

**Claims Adjusters**

They hand off raw incident notes and policy dates to the AI to build organized claim histories.

**Compliance Officers**

They use the tool to check if claims meet the reporting requirements set by policy milestones.

**Insurance Auditors**

They use the timeline tools to verify the accuracy and sequence of claim processing.

## What Changes When You Connect

01 Converts disconnected timestamps into a single chronological sequence.

02 Automates the verification of reporting windows against policy rules.

03 Reduces manual sorting by filtering timelines by specific event types.

04 Provides instant narrative summaries of time elapsed since an incident.

## Real-World Applications

**Reporting Compliance Audits**

Check if a claimant reported an incident within the mandatory window defined in their policy.

**Claim History Reconstruction**

Turn a list of messy, unorganized incident notes into a clean, ordered timeline.

**Timeline Categorization**

Isolate only the policy milestones or only the witness statements from a long claim history.

**Aging Analysis**

Get a quick summary of how many days have passed since an incident occurred to track claim aging.

## Patterns to Avoid

### Manual timeline reconstruction

AVOID

An adjuster spends hours copying dates from various PDF notes into an Excel sheet to see the claim flow.

✓ INSTEAD

Feed the raw incident data directly to your agent and use `get_chronological_timeline` to build the sequence instantly.

### Broad compliance guessing

AVOID

A compliance officer looks at a claim and assumes it was reported on time based on a general feeling.

✓ INSTEAD

Run `validate_reporting_compliance` to check the report date against specific policy milestones for an objective answer.

### Information overload

AVOID

Trying to find a specific witness statement within a massive, thousand-line claim history log.

✓ INSTEAD

Use `filter_timeline_by_type` to pull out only the specific event categories you need to see.

## The Right Fit

Insurance professionals who handle high volumes of messy claim data should connect this MCP. If your claim history is already perfectly structured in a clean database, you likely won't need it. The decision comes down to whether you spend more time reading notes than actually adjusting claims.

# Incident to Claim Timeline MCP Tools

These tools transform scattered incident data into structured, chronological histories and compliance reports.

| # | TOOL | DESCRIPTION |
|---|---|---|
| 01 | `summarize_claim_status` | This tool provides a high-level narrative of the time elapsed between the initial incident and the current status. |
| 02 | `filter_timeline_by_type` | Use this to extract a specific subset of the timeline, such as only notices or only policy milestones. |
| 03 | `get_chronological_timeline` | This tool reconstructs a complete, ordered history using all the incident-related data points you provide. |
| 04 | `validate_reporting_compliance` | This tool checks if a claim was reported within the specific timeframes required by policy milestones. |

# See It in Action

Real prompts you can use once this MCP is connected to your AI agent through Vinkius Cloud.

**U** Create a timeline from these events: Incident on 2023-01-01 (Burst Pipe), Policy Effective on 2022-12-01, and Notice sent on 2023-01-05.

1. 2022-12-01: Policy Effective (Policy)
2. 2023-01-01: Burst Pipe (Incident)
3. 2023-01-05: Notice sent (Notice)

**U** Was the claim reported within 3 days of the incident? Incident: 2023-05-10, Notice: 2023-05-12.

The claim is Compliant. The notice was reported 2 days after the incident.

**U** Summarize the status for a timeline where the incident was 2023-06-01 and today is 2023-06-10.

Total days since incident: 9. The incident occurred 9 days ago.

# Frequently Asked Questions

01 **What can this MCP do with my insurance data?**

It sequences incident events, witness statements, and policy milestones into a single, ordered timeline. It can also check for reporting compliance and summarize the time elapsed since an incident.

02 **Which AI clients can I use with this MCP?**

You can use this MCP with any MCP-compatible client, including Claude, Cursor, Windsurf, and VS Code.

03 **How does it check for compliance?**

The tool compares the date an incident was reported against the required timeframes defined in your policy milestones.

04 **Can I filter the timeline for specific events?**

Yes, you can use the filter tool to isolate specific categories like notices or policy milestones from the full history.

05 **Do I need to host this myself?**

No, Vinkius hosts and manages the MCP for you. You just connect your client and start using the tools.

06 **How do I create a chronological sequence of events?**

Use the `get_chronological_timeline` tool by providing the incident events, witness statements, notices, and policy milestones as JSON strings.

07 **Can I check if a claim was reported on time?**

Yes, use the `validate_reporting_compliance` tool to compare the earliest incident event against the earliest notice based on your specified deadline.

08 **How can I get a summary of the time elapsed since the incident?**

You can use `summarize_claim_status` with the generated timeline and a target date to receive a narrative summary and total days elapsed.

# Go Live in 60 Seconds

Get your connection token from **cloud.vinkius.com**, then paste the endpoint URL into any MCP-compatible client.

YOUR MCP ENDPOINT

```
https://edge.vinkius.com/[TOKEN]/mcp
```

| CLIENT | WHERE TO CONFIGURE |
|---|---|
| Claude AI | Profile → Customize → Connectors → "+" → Add custom connector → Paste endpoint |
| Cursor | Settings → Features → MCP Servers → "+ Add New MCP Server" → Type: SSE → Paste endpoint |
| VS Code | Ctrl/Cmd+Shift+P → "MCP: Add Server" → add `"incident-to-claim-timeline": { "url": "…" }` |
| Windsurf | MCP Settings → `mcp_settings.json` → Add endpoint URL |
| ChatGPT | Settings → Tools & plugins → Add MCP server → Paste endpoint |
| Gemini | Extensions → Add MCP Server → Paste endpoint URL |

ASK AN AI ABOUT THIS

Let your preferred AI explain this MCP server

- Ask ChatGPT
- Ask Claude
- Ask Perplexity
- Ask Gemini
- Ask Grok

READY TO CONNECT

# Incident to Claim Timeline is live on Vinkius Cloud.

Get your connection token, paste it into your AI agent, and start building. No SDK. No deployment. Just results.

**Start at cloud.vinkius.com →**

vinkius.com · support@vinkius.com

## INDEPENDENT PLATFORM DISCLAIMER

Vinkius is an independent platform and is not affiliated with, endorsed by, sponsored by, verified by, or otherwise authorized by Incident to Claim Timeline. All third-party trademarks, logos, and brand names are the property of their respective owners. Their use in this document is strictly for informational purposes to identify service compatibility and interoperability.

## DOCUMENT INFORMATION

| | |
|---|---|
| Generated | September 2026 |
| MCP Server | Incident to Claim Timeline MCP |
| Server ID | `01a0e04d-be87-715f-b8ce-b0fd48db95fd` |
| Platform | Vinkius Cloud for AI Agents |
| Endpoint | `https://edge.vinkius.com/{token}/mcp` |

## LICENSE & USAGE

This document is generated automatically by the Vinkius PDF Engine. Content reflects the MCP server configuration at the time of generation and may change as updates are deployed. For the most current information, visit vinkius.com/en/ai-agent-connect/incident-to-claim-timeline.