Compliance and SIEM

Evidence packs, SIEM export, retention, EU AI Act readiness.

Overview

Compliance & SIEM

Evidence pack

The **Compliance** screen assembles an auditor-ready evidence pack for any period: the policies in force (MCP default + rules, detector actions), risk sensitivity, capture settings, top models/tools/MCP servers, and activity summaries. Export and hand it to your auditor — this is the artifact the EU AI Act's transparency expectations (August 2026) map onto.

SIEM export

Pull or push your audit events into your own SIEM:

- **Pull**: an OCSF-formatted API your collector polls with a scoped credential.
- **Push**: the platform delivers events to your HTTPS endpoint.

Either way the data is yours — Sentilai's audit trail is designed to feed your existing security operations, not replace them.

Retention

Audit metadata retention is per-tenant (default 14 days) with automated purge; device logs are fixed 14-day; captured prompts (opt-in) follow the same tenant retention. Support tickets persist for the support relationship. Deletion on offboarding is contractual and scripted.

Egress blocklist

The Policy page generates a network **egress blocklist** from what's actually configured for your tenant — the direct provider hosts to block and the one Gateway host to allow — so your network team can enforce "AI only via Sentilai" at the firewall.

Generate an evidence pack

Compliance produces a point-in-time record of how AI is governed in your organization: the active policy, who had access, and what the Gateway actually enforced.

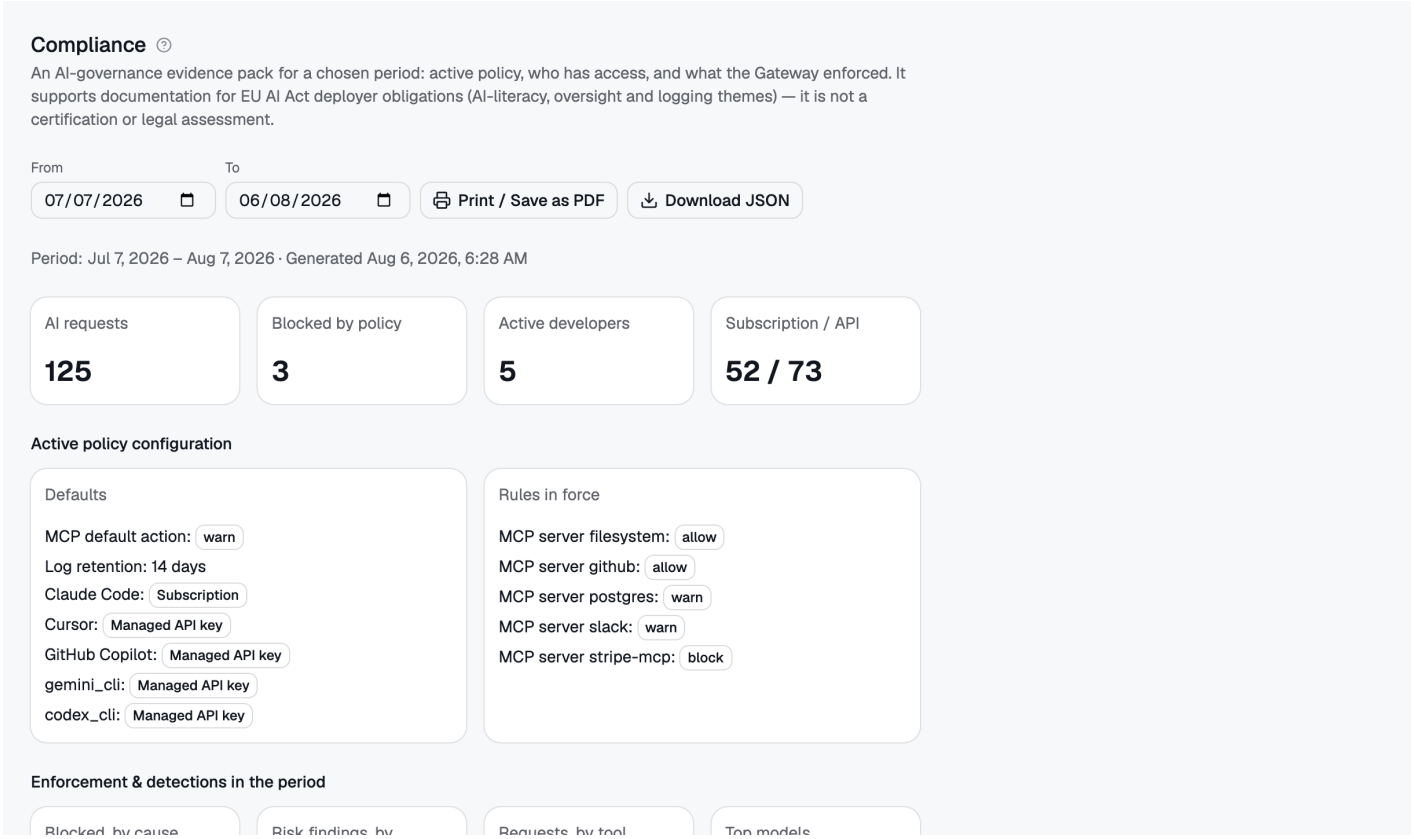
1. Pick a **From** and **To** date.
2. **Print / Save as PDF** for a document to hand over, or **Download JSON** for a machine-readable copy.

What it contains

- **Totals** for the period: AI requests, blocked by policy, active developers, and the subscription/API split.
- **Active policy configuration** — defaults, retention, per-tool connection modes, and every MCP rule in force.
- **Enforcement and detections** — blocks by cause, risk findings by detector, requests by tool, top models.
- **Access inventory** — seats, admins, developers, and registered devices.

What it is, and isn't

It supports documentation for EU AI Act deployer obligations — the AI-literacy, oversight and logging themes — by showing what you actually did. It is **not** a certification, and it is not a legal assessment. Treat it as evidence you can hand to an auditor, not as an answer to whether you comply.



The Compliance evidence pack: totals, active policy, enforcement and access inventory for a period.

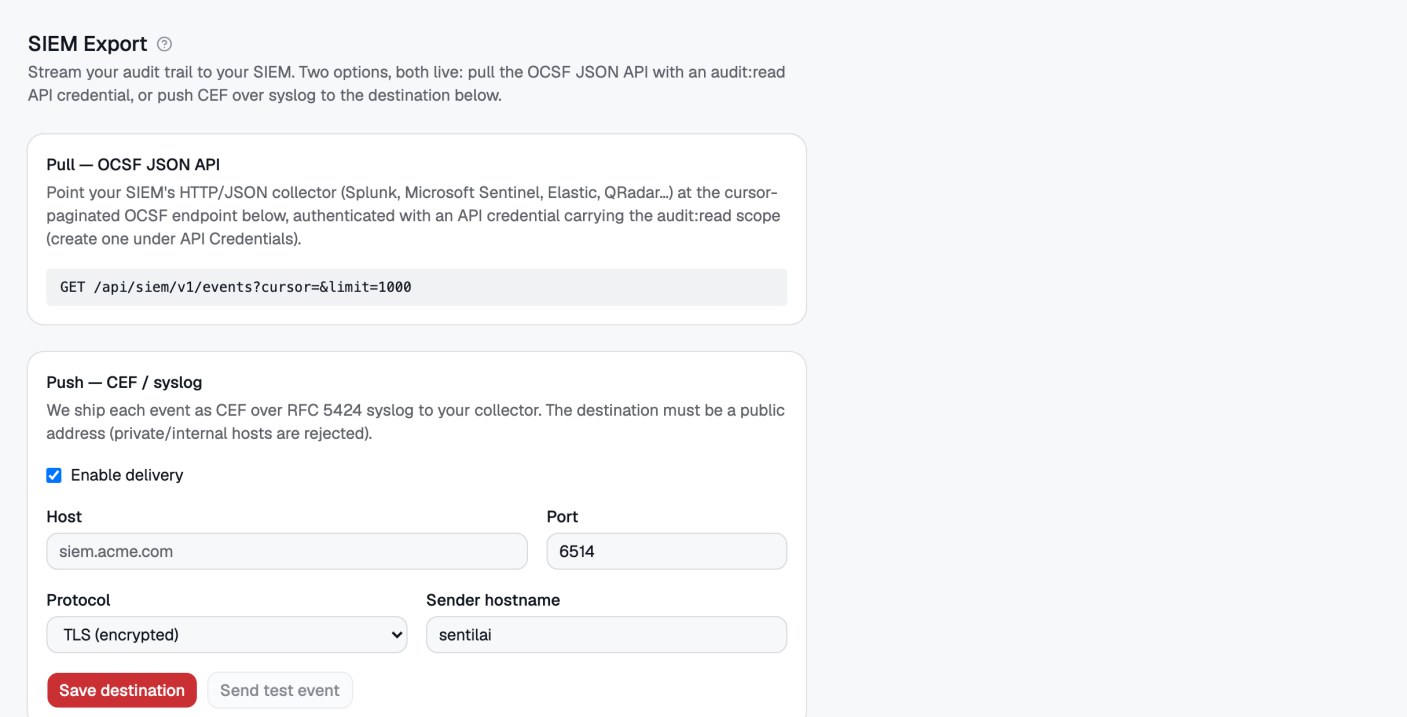
Stream events to your SIEM

SIEM Export pushes Sentilai events to your security monitoring stack as they happen.

1. Enter the destination host and port, and pick the protocol (`tcp` or `tls`).
2. **Save destination**.
3. **Send test event** and confirm it lands in your SIEM.

Destinations must be publicly resolvable addresses — an internal-only host will be rejected, since our Gateway has to reach it.

Prefer `tls` unless the destination genuinely cannot terminate it: these events describe your developers' AI usage and shouldn't cross the internet in the clear.



SIEM Export — the OCSF pull API and the CEF/syslog push destination, with live delivery status.

What the evidence pack contains

Compliance produces a point-in-time record for a date range you choose, as a printable report and as JSON.

The four parts

Traffic — AI requests, how many were blocked by policy, how many developers were active, and the split between subscription and API-key usage.

Active policy configuration — the MCP default action, your retention setting, per-tool connection modes, and the **rules in force**: every risk-detector rule and every MCP server rule. If you have changed nothing, it says "code defaults only", which is itself an honest and useful statement.

Enforcement and detections — what was blocked and why, risk findings by detector, requests by tool, and the top models.

Access inventory — seats used against your cap, how many devices are registered, and a table of every admin and developer with their role and enrolment date.

Two formats

Print / Save as PDF produces a report with a proper header and footer, for attaching to something. **Download JSON** gives you the same data as `senticons-evidence-<date>.json` for your own tooling.

What it is not

The report says so itself: it supports documentation of **EU AI Act deployer obligations** — AI literacy, human oversight, logging — but it is **not a certification and not a legal assessment**. It is evidence that you have controls and that they were operating. Whether that satisfies a particular obligation is a question for your counsel.

We would rather say this plainly than let a customer discover it in an audit.

Using it in an audit

Some practical notes from what auditors actually ask.

Generate it at the time, not afterwards

Retention is finite. An evidence pack covering March, generated in March, keeps its detail; one generated in December may cover a window whose events have been purged.

Generate one per quarter and keep it. It takes a minute and it converts a perishable dataset into a durable document.

Answering the common questions

"How do you know which AI tools your developers use?" — Requests by tool, plus the device inventory.

"How do you control what data goes to AI providers?" — The risk-detector rules in force, with the detection counts showing they were operating rather than merely configured.

"Who has access?" — The access inventory, with roles and enrolment dates.

"What happens when someone leaves?" — Offboarding revokes devices and removes passkeys in one action; the device inventory across two consecutive packs shows it happened.

The gap to be honest about

The pack shows what happened **through Sentilai**. A developer using a personal API key on a personal machine does not appear, because they never touched the Gateway. If an auditor asks about coverage, the honest answer is that you govern the managed path and monitor for drift — and that the device inventory plus the non-compliant flag is how you detect machines slipping out of it.

Claiming complete coverage of something you cannot see is how organizations get into trouble with auditors, not out of it.

SIEM: pull with the OCSF API

The pull option: your SIEM asks Sentilai for events on its own schedule.

The endpoint

`GET /api/siem/v1/events?cursor=&limit=1000`

Events are in **OCSF** — the Open Cybersecurity Schema Framework — so they land in a shape Splunk, Microsoft Sentinel, Elastic and QRadar already understand.

Authentication

An **API credential** with the `audit:read` scope. Create it on **API Credentials**; the secret is shown exactly once.

Give this credential only `audit:read`. It is going to live in a configuration file in another system, and it should be able to do nothing except read audit events.

Cursor pagination

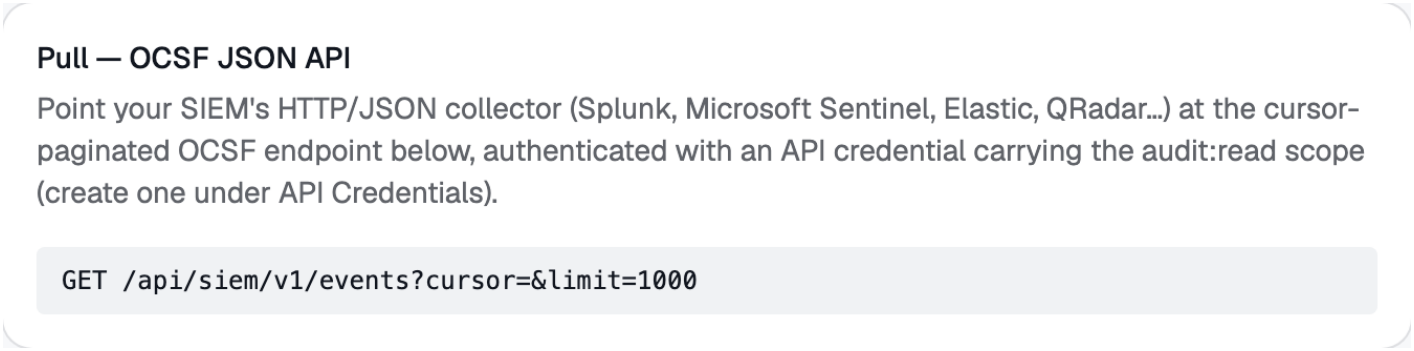
Each response carries a cursor. Store it, pass it next time, get what has happened since. This is what makes the integration resumable: a SIEM that was down for a day catches up rather than losing the gap.

Choosing pull or push

Pull is more reliable — your SIEM controls the pace and can retry. It is the default recommendation.

Push is lower-latency and suits SIEMs that prefer to receive syslog. See *SIEM: push over syslog*.

You can run both.



The pull card on SIEM Export — the OCSF endpoint, and the one-scope credential it needs.

SIEM: push over syslog

The push option: Sentilai sends events to your collector as they happen.

Configuring

SIEM Export ? push:

- **Enable delivery**
- **Host** and **Port** (6514 by default)
- **Protocol** — TLS encrypted, or TCP plaintext
- **Sender hostname** — how the events identify themselves in your SIEM

Use TLS. Plaintext exists for collectors inside a network you already trust; audit events describing your AI traffic are not something to put on the wire in the clear.

Restrictions

One destination per organization. Private and internal hostnames are rejected — the collector must be reachable from Sentilai, so a `10.x` address will not work. Terminate TLS on something with a public name, or use the pull API from inside your network instead.

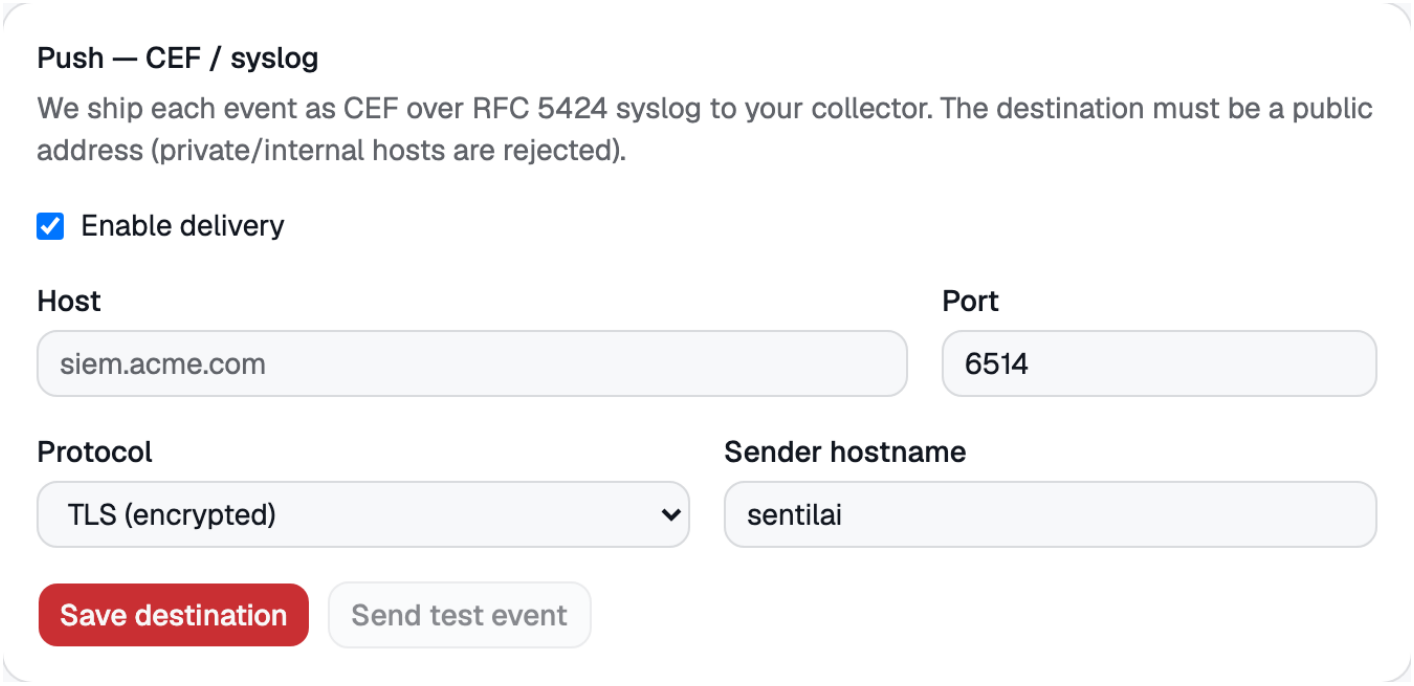
Test it

Send test event delivers a `siem_push_test` event. Look for it in your SIEM before assuming the integration works.

Delivery status

The panel shows the delivered count, last success, last attempt, and the last error. When push stops working — an expired certificate, a moved collector — this is where it shows up. The status badge reads **Active** or **Paused**.

Check it occasionally. Silent failure of an audit pipeline is the failure mode that matters most and announces itself least.



The push card on SIEM Export — syslog destination, TLS, and live delivery status with the last error if one occurred.

What we send to your SIEM

The full list of events Sentilai produces, what each one means, and which fields carry it.

Everything below arrives through **both** delivery modes — the OCSF pull API and the CEF/syslog push. They are the same stream in two formats, not two different feeds.

AI request events

One event per AI request your developers make through the Gateway.

Event	Meaning
`llm_request`	a proxied AI request to any provider
`mcp_call`	an MCP tool call
`mcp_observation`	an MCP tool definition changed on a device
`siem_push_test`	the synthetic event behind **Send test event**

A request that triggered a detector, or that policy blocked, arrives as an **OCSF Detection Finding (class 2004)** — the class your SIEM already routes to an analyst. Everything else is **API Activity (6003)**, the routine "this happened" record.

Each carries: who (user, device), what tool and model, which provider, token counts, duration, outcome, and — when something fired — every detector, every specific finding, and **why it was blocked**.

Ungoverned AI agent events

Autonomous AI agents installed on developer machines that do **not** route through Sentilai.

Event	Meaning
`ungoverned_agent.detected`	an ungoverned agent appeared on a device
`ungoverned_agent.resolved`	it is gone — the developer removed it
`ungoverned_agent.access_blocked`	your policy withdrew that device's AI access
`ungoverned_agent.access_restored`	access came back, because the agent was removed

Only the transitions are sent. A device that has had the same agent for a week produces one event, not one every thirty seconds.

The event names the agent and the specific risks found — including `openclaw.lethal_trifecta`, which means one agent has host command execution, access to private data, and a channel to the outside world at the same time. That is the one to alert on.

Administrator actions

Every change an administrator makes in the console, as an **OCSF Entity Management (3004)** event: policy changes, API credentials created or revoked, devices revoked, developers invited and offboarded, provider keys, SSO configuration, alert channels, and changes to the SIEM destination itself.

Each says who did it (name and email), what they changed, and what the new value is.

Worth a rule of its own: `siem_destination.updated` and `siem_destination.deleted` mean somebody changed where your audit evidence goes.

Failed attempts are not recorded as changes — a rejected request changed nothing, and an audit log listing things that never happened is worse than one that omits attempts.

What we never send

- **Prompt and response content.** Events are metadata. Captured conversations, if you enable them, stay in Sentilai and are never part of this stream.
- **The secret itself.** A finding tells you an AWS key was present and how many times — never the key.
- Provider API keys, tokens, or credentials of any kind.

Which threats we detect

The signals that can appear on an event, what each one means, and where you control it.

In the request content

Signal	What it finds
Secret scanning	AWS keys, GitHub tokens, private keys, Anthropic/OpenAI keys, Slack tokens, Google API keys, JWTs
Personal data (PII)	IBANs, payment cards, phone numbers, email addresses — each checksum-validated, so a random 16-digit number is not reported as a card
Credentials in context	phrasing that suggests a credential is being discussed even when no key matches
Prompt injection	our classifier's verdict, with its score
Dependency risk	packages the AI suggested that do not exist, are known-vulnerable, are suspiciously new, or match a known slopsquat
MCP rug-pull	an MCP tool that changed its definition after you approved it
Lethal trifecta	one MCP call combining private data, untrusted content and a way out

All of these are set on **Policy**, each to `allow`, `report`, `warn` or `block`.

On the device

The Endpoint Suite scans for ungoverned AI agents and reports what it finds: whether one is installed, whether it starts automatically, whether its gateway is reachable from the network, whether it requires authentication, whether it can run commands unsandboxed, whether anyone who can message it can drive it — and whether all three legs of the lethal trifecta are present.

We **detect** ungoverned agents; we do not govern their traffic. They talk to their providers directly and we are not in that path. What the `block` policy does is take away the developer's *governed* tools until the agent is gone.

What "blocked" tells you

When a request is blocked, the event names the cause:

- `classifier` — the prompt-injection classifier refused it
- `risk` — a content detector set to `block` fired
- `mcp_policy` — an MCP server or tool rule refused the call

Retention, and what it means for your SIEM

Events are kept for your organization's retention period and then deleted. The stream is gap-free **within that window**: a collector that catches up after an outage gets everything it missed, as long as the outage was shorter than your retention setting.

If your SIEM stops receiving events for longer than that, the gap is permanent. This is why the delivery status on the SIEM Export screen is worth an occasional look — a stalled audit pipeline is the failure that announces itself least.