

Sentrilite — Threat Detection-as-Code (DAC), eBPF-based, Observability, Runtime-Security & Cloud-Security Posture Management in One Platform with AI/LLM Insights.

🌟 Quick Trial

Run it with Pre-built Docker Image:

On Your linux server, run:

```
sudo docker run \
  --name sentrilite \
  --privileged \
  --network host \
  -v /sys/fs/bpf:/sys/fs/bpf \
  -v /sys/kernel/debug:/sys/kernel/debug \
  sentrilite/local:1.0.0
```

In a web browser, open: <http://localhost:8080>. PDF Reports can be downloaded and a new report is generated every 5 mins.

Run it on Kubernetes Cluster

```
git clone https://github.com/sentrilite/sentrilite.git . cd
sentrilite/charts
```

```
helm upgrade --install sentrilite charts/sentrilite -n kube-system
--create-namespace
```

Sentrilite Alert Report

Sentrilite Alert Report

Generated at:

Combined Alerts Distribution



Risk Color Legend

- Critical / High risk – immediate triage
- Medium risk – monitor & investigate
- Informational / low risk

Alert Breakdown

High Risk: 558
Medium Risk: 684
Low Risk: 637
Total: 1879
Total alerts (all nodes): 1879
Responding nodes: 3

Tags Summary (top 10):

network: 1238
privilege-escalation: 487
network-policy-change: 454
kernel: 58
firewall: 53
file: 24
permissions: 24
scanner: 19
internal: 10
error: 10

Top Processes / Commands

bash (1373)
/usr/sbin/iptables (258)
/usr/sbin/ip6tables (192)
/usr/bin/sudo (19)
package-vuln: trivy not installed or not in PATH (10)

Top Source IPs

No IPs with count > 5.

Node Risk Overview

ec2 js-east-2.compute.amazonaws.com
ec2 .compute-1.amazonaws.com
ec2 .compute-1.amazonaws.com

168	246	619
195	219	9
195	219	9

Main Dashboard

Sentrilite: Hybrid-Cloud Observability & Security

Download PDF Report

Download Combined Alerts (JSON)

Choose File

node_list.txt

Upload Node List

Download Dashboard

Select All

Clear All Alerts

Create Rule

match_key (e.g. cmd)

match_values (comma sep)

tags (comma separated)

risk_level

server_tag (default: all)

Apply to Selected

View Rules

Delete Rules

Delete All Rules

Network Rule

Select	Server IP	Status	Alerts	Groups	Dashboard	AI Insights
<input type="checkbox"/>	ec2-3-17-135-143.us-east-2.compute.amazonaws.com	Online	Critical	private	Open	View
<input type="checkbox"/>	ec2-3-86-227-160.compute-1.amazonaws.com	Online	Critical	aws	Open	View
<input type="checkbox"/>	ec2-54-157-205-225.compute-1.amazonaws.com	Online	None	aws	Open	View
<input type="checkbox"/>	myapp-eastus-001.cloudapp.azure.com	Unreachable	Unknown	azure	Open	View
<input type="checkbox"/>	myapp-eastus-002.cloudapp.azure.com	Unreachable	Unknown	azure	Open	View
<input type="checkbox"/>	gke-node-01.us-central1.example.internal	Unreachable	Unknown	gcp	Open	View
<input type="checkbox"/>	gke-node-02.us-central1.example.internal	Unreachable	Unknown	gcp	Open	View

Live Server Dashboard

EDR Manager

+ Add New Rule

View Rules

Delete All Rules

Clear Events

XDR Manager

+ Add New Rule

View Rules

Delete All Rules

Sentrilite Live System Events Dashboard

Resume

Alerts On

Alert History

Connected

High Risk: 6

Medium: 0

Low: 20

Filter UID/username

Filter IP

Filter CMD

Filter TAG

Live Events

[2025-10-26T20:34:20.743Z] PID=261130 UID=0 USER=root CMD=php-fpm3.3 IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:34:22.253Z] PID=458492 UID=1000 USER=ubuntu CMD=bash CMD=/usr/bin/nc ARG=1 IP=127.0.0.1 TYPE=EXECVE [scanner, suspicious-network, lateral-movement]

[2025-10-26T20:34:24.001Z] PID=458492 UID=1000 USER=ubuntu CMD=nc IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:34:24.001Z] PID=201530 UID=110 USER=chrony CMD=chronyd IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:34:25.454Z] PID=458493 UID=1000 USER=ubuntu CMD=bash CMD=/usr/bin/cat ARG=/etc/passwd IP=127.0.0.1 TYPE=EXECVE [info-disclosure, info-disclosure]

[2025-10-26T20:34:30.745Z] PID=458493 UID=1000 USER=ubuntu CMD=systemd-logind IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:34:32.242Z] PID=458494 UID=1000 USER=ubuntu CMD=bash CMD=/usr/bin/cat ARG=/etc/passwd IP=127.0.0.1 TYPE=EXECVE [info-disclosure, info-disclosure]

[2025-10-26T20:34:40.086Z] PID=458495 UID=1000 USER=ubuntu CMD=bash CMD=/usr/bin/sudo ARG=ls IP=127.0.0.1 TYPE=EXECVE [privilege-escalation]

[2025-10-26T20:34:40.088Z] PID=458495 UID=1050 USER=ubuntu CMD=sudo IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:34:40.090Z] PID=461105 UID=991 USER=systemd-resolve CMD=systemd-resolve IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:34:40.090Z] PID=458497 UID=0 USER=root CMD=sudo CMD=/usr/bin/ls IP=127.0.0.1 TYPE=EXECVE [privilege-escalation, privileged-user-activity]

[2025-10-26T20:34:40.090Z] PID=458379 UID=0 USER=root CMD=trace_events IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:34:40.238Z] PID=201530 UID=110 USER=chrony CMD=chronyd IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:34:40.745Z] PID=261130 UID=0 USER=root CMD=php-fpm3.3 IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:34:47.454Z] PID=458498 UID=1000 USER=ubuntu CMD=bash CMD=/usr/bin/sudo ARG=nc IP=127.0.0.1 TYPE=EXECVE [privilege-escalation]

[2025-10-26T20:34:47.462Z] PID=458498 UID=1000 USER=ubuntu CMD=sudo IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:34:47.611Z] PID=261105 UID=991 USER=systemd-resolve CMD=systemd-resolve IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:34:47.619Z] PID=458500 UID=0 USER=root CMD=nc CMD=/usr/bin/nc ARG= IP=127.0.0.1 TYPE=EXECVE [scanner, privilege-escalation, suspicious-network, lateral-movement]

[2025-10-26T20:34:47.619Z] PID=458500 UID=0 USER=root CMD=nc IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:34:50.747Z] PID=261130 UID=0 USER=root CMD=php-fpm3.3 IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:34:50.450Z] PID=201530 UID=110 USER=chrony CMD=chronyd IP=127.0.0.1 TYPE=SOCKET

[2025-10-26T20:35:00.749Z] PID=261130 UID=0 USER=root CMD=php-fpm3.3 IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:35:01.535Z] PID=458501 UID=0 USER=root CMD=nc IP=127.0.0.1 TYPE=SOCKET [privilege-escalation, privileged-user-activity]

[2025-10-26T20:35:01.539Z] PID=458502 UID=0 USER=root CMD=nc CMD=/usr/bin/nc ARG= IP=127.0.0.1 TYPE=EXECVE [privilege-escalation, privileged-user-activity]

[2025-10-26T20:35:01.540Z] PID=458503 UID=0 USER=root CMD=nc CMD=/usr/lib/synstat/debian-ai ARG=1 IP=127.0.0.1 TYPE=EXECVE [privilege-escalation, privileged-user-activity]

Description

Sentrilite is a Detection-as-Code (DAC), Hybrid-Cloud Programmable Observability, Runtime-Security & CSPM Platform and streams structured, real-time events to a web UI where custom rules drive risk scoring, alerting, and reporting.

Hybrid & multi-cloud ready: Works the same across public clouds and on-prem—EKS, GKE, AKS, vanilla Kubernetes, bare-metal, and edge—so you get a consistent, low-overhead security and observability layer for hybrid/multi-cloud environments all managed from a single dashboard.

In Kubernetes, Sentrilite runs as a privileged DaemonSet on every node (no changes to your workloads). Each agent uses hostPID/hostNetwork to observe container processes, then enriches events with pod metadata (namespace, pod, container, UID) by correlating cgroups with the API server. This lets you see all the activity at the container/pod level:

- Detection-As-Code: Add / modify detection rules instantly via json files. Hot Reload — no rebuilds, no redeloys. See `custom_rule.json` and `security_rules.json`
- Seamless install & upgrade: `kubectl` apply the DaemonSet (`sentrilite.yaml`) and you're done; rolling updates pick up new rules and images cluster-wide.
- Process visibility: Capture commands and args (`execve`) per container/pod with user, PID/PPID, and image context; trigger rules like “alert on cat `/etc/passwd`” or “block high-risk binaries.”
- File activity: Match sensitive file paths (config, keys, tokens) using rule packs; flag exfil patterns and privilege-escalation attempts.
- Network activity: Trace socket opens and outbound/inbound connects/binds; create rules for destinations IP, ports, or CIDRs (e.g., “deny egress to `1.2.3.0/24:443`”).
- Live operations UI: Watch streaming events per node/pod, plus live node/server health and OOMKilled notices; filter by namespace/pod/container in real time.
- Custom rules & risk: Declarative JSON rules tag and score events; high-risk findings become alerts with clear, human-readable summaries that include k8s context.
- Reporting: Generate rich summary reports (e.g., PDF/CSV) showing timelines, risky commands, and per-namespace insights for audits and incident reviews.
- Real-time security posture: Optional controls (like iptables-backed allow/deny rules) help you respond quickly to suspicious network behavior.
- Third-Party-Integrations: Seamlessly integrate with external alerting tools like: `prometheus:alert-manager`, `pagerduty` etc.
- LLM-powered insights: automatically summarize trends, explain anomalies, and suggest remediation/rules from live telemetry and alerts.

In summary, Sentrilite gives you container-aware process, file, and network visibility with minimal overhead, live dashboards for fast triage, and exportable reports for compliance and forensics—all from a single, node-level DaemonSet.

- Website: <https://sentrilite.com>
- Email: info@sentrilite.com
- Hybrid Cloud Demo: <https://youtu.be/FmFUs0ZhdIY>
- Linux (bare-metal) Demo: <https://youtu.be/rRexG-f6YFM>

✨ Key Features

- Multi-Cloud/On-Prem visibility and management from a single dashboard.
- eBPF syscall & network visibility
- Real-time dashboards (Nginx + WebSocket server)
- Custom rules with risk scoring and alerting
- Kubernetes enrichment (namespace/pod/container/UID) when running as a DaemonSet
- OOMKilled alerts and pod watchers (best effort if K8s APIs available)
- Seamlessly integrate with prometheus alert-manager/pagerduty

📦 Contents of this Bundle

File	Purpose
<code>trace_syscall.o</code>	eBPF kernel object for syscall monitoring
<code>install.sh</code>	Script to load the ebpf kernel module
<code>unload_bpf.sh</code>	Script to unload the ebpf kernel module
<code>trace_events</code>	Userspace program for network/socket activity
<code>sentrilite</code>	Go websocket server that forwards live events to browser dashboard
<code>main.html</code>	Main frontend UI for viewing node status
<code>dashboard.html</code>	Local frontend UI for viewing live events
<code>sys.conf</code>	Configuration file

<code>custom_rules.json</code>	Custom Ruleset. Default 30+ rules. Can be extended at runtime. Refer the Product Guide for details.
<code>security_rules.json</code>	Linux & Kubernetes Security Rules. Can be extended at runtime. Refer the Product Guide for details.
<code>sensitive_files.json</code>	Files to Monitor. Refer the Product Guide for details.
<code>sentrilite.yaml</code>	Sentrilite daemonset manifest to install on Kubernetes cluster
<code>kustomization.yaml</code>	Kubernetes fest to update License.key
<code>charts</code>	Helm Charts for installation
<code>bpftool</code>	Tool to load and attach kernel tracepoints. Source: https://git.kernel.org/pub/scm/linux/kernel/git/bpf/bpf-next.git
<code>LICENSE.bpftool</code>	GPL-2.0 License for bpftool. Source: https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/plain/LICENSES/preferred/GPL-2.0
<code>license.key</code>	Sentrilite License key file
<code>LICENSE.txt</code>	Sentrilite License Agreement
<code>install.README</code>	This installation guide
<code>dashboard.README</code>	Dashboard usage guide
<code>Product Guide v1.1.pdf</code>	Sentrilite Product Guide

System Requirements

- Linux Kernel with eBPF support (Linux 5.8+ recommended)
- Root privileges (for loading eBPF programs)
- Ports: 80 (dashboard), 8765 (WebSocket)
- Kubernetes (optional): Cluster access with ability to run a privileged DaemonSet

General Requirements

- bpftool: Load eBPF programs and manage maps `sudo apt install bpftool` (Ubuntu)
- libbpf & headers Required by the kernel loader (`trace_events`) Pre-installed on most modern distros (use bundled binary)
- nginx Required to view dashboard `sudo apt install nginx`

Licensing

The project is currently using a trial license.key .

Third-Party Integrations (PagerDuty & Alertmanager)

- Kubernetes: add URLs in a ConfigMap (e.g., `ALERTMANAGER_URL`, optional `PAGERDUTY_EVENTS_URL`) and the PagerDuty routing key in a Secret (`PAGERDUTY_ROUTING_KEY`).
- Standalone Linux: set the same keys in `sys.conf` (e.g., `ALERTMANAGER_URL=http://...`, `PAGERDUTY_ROUTING_KEY=...`; PD URL defaults to US if omitted).

Sentrilite prefers env vars (K8s) and falls back to `sys.conf` (bare metal).

Note: Alertmanager must be reachable and supports v2 API (`/api/v2/alerts`). PagerDuty uses Events v2.

Alerts & K8s Enrichment

- Events include (when available): `k8s_namespace`, `k8s_pod`, `k8s_container`, `k8s_pod_uid`.
 - OOMKilled alerts and pod watchers run best-effort when the agent can access K8s APIs.
-

Un-installation Steps

For Kubernetes Cluster: EKS/AKS/GKE or Private Kubernetes Cluster

```
kubectl -n kube-system delete ds/sentrilite-agent
```

(Optional) If pods hang in Terminating:

```
kubectl -n kube-system delete pod -l app=sentrilite-agent --force --grace-period=0
```


For Non-Kubernetes Linux based Cluster

Run the following commands as root.

```
sudo ./unload_bpf.sh
```

Support

For licensing, troubleshooting, or feature requests:

-  info@sentrilite.com
-  <https://sentrilite.com>