Sentinel by CryptoITData

Your server is under attack right now. See by whom.

A security agent that runs 24/7 on a single Linux server: it collects from six sources, raises incidents, puts the block button on your phone, scans for vulnerabilities and generates patch procedures with a verified way back.

The figures below are real, from a production install — AlmaLinux 9, 14 monitored applications, the last 24 hours.

Detects

Six collection sources, deterministic rules every 10 seconds, incidents deduplicated by fingerprint.

Alerts

The incident on Telegram in seconds, with the block button under it and an AI verdict next to the deterministic one.

Repairs

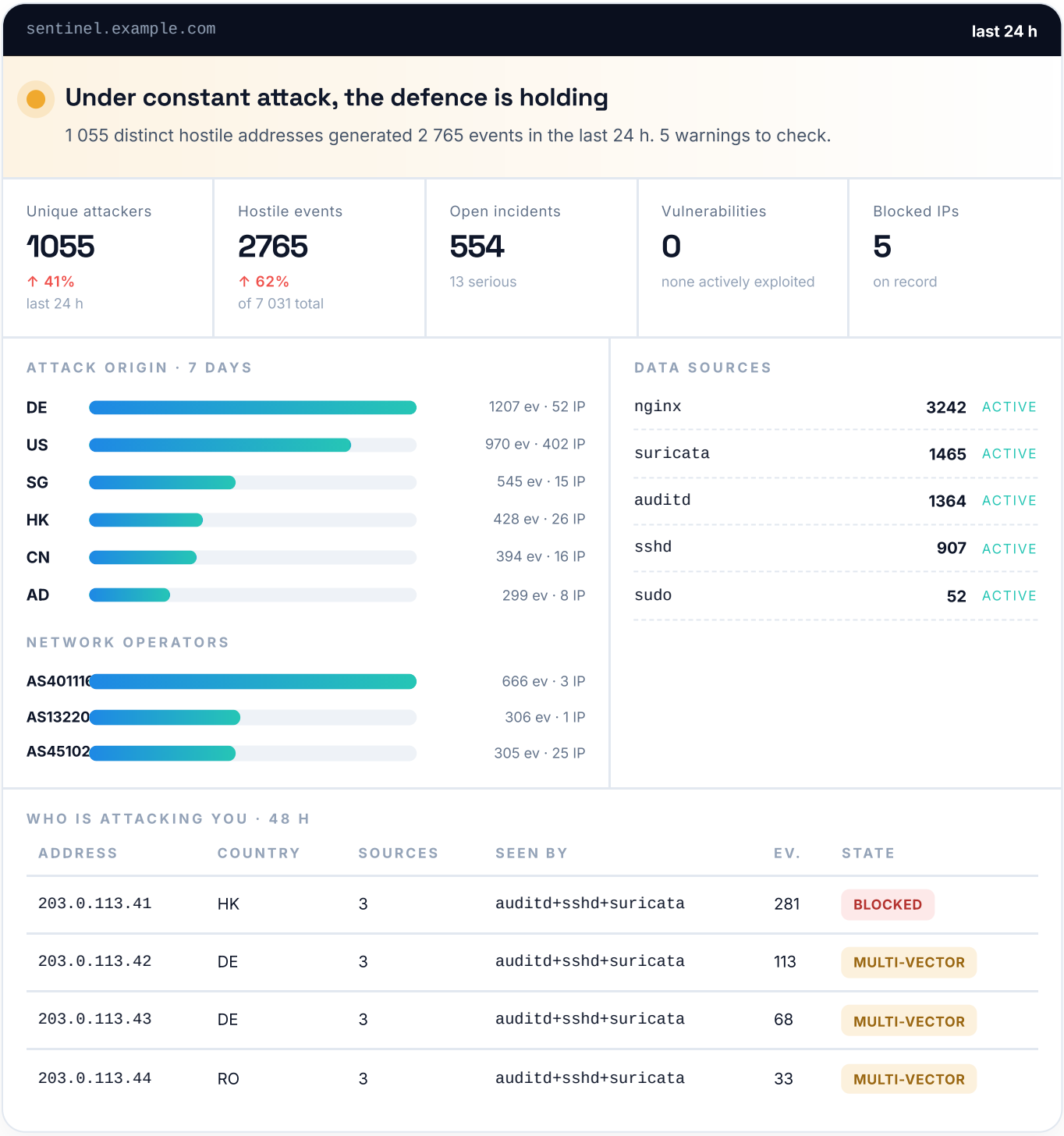
Patch plans prioritised by KEV and exposure, with a verified backup and automatic rollback.

1 055	2 765	6	0	14
unique attackers / 24h	hostile events	collection sources	open vulnerabilities	active services

THE SECURITY DASHBOARD

The first line answers a single question: am I OK?

Reading order is the design. The verdict, then the numbers behind it, then the interpretation, then the tables it came from. Anyone who reads only the first line already has the right answer.



The real dashboard, with today's production data. "Multi-vector" means the address was seen by three independent collectors — a mass scanner touches a single surface; whoever tries three has chosen you.

INTERPRETATION

A number is not an observation.

“2 765 events” is a figure. “root was targeted 1 086 times from 93 addresses — disable password authentication for root” is an observation. The engine runs deterministic rules over the data and produces only things you can act on.

5 addresses are attacking you on several fronts at once

They appear in 3 independent sources in the last 48 h. A mass scanner touches a single surface; whoever tries three has chosen you.

TO DO `/block 203.0.113.41 24h`

root is target #1 — 1 086 attempts from 93 addresses

As long as SSH password authentication for root is possible, every one of those attempts has a chance.

TO DO `grep PermitRootLogin /etc/ssh/sshd_config`

Concentrated attack infrastructure: Netiface America, Inc.

666 hostile events from just 3 addresses in the same network. That is rented attack infrastructure, not scattered compromised machines — and the next address will come from the same place.

5 blocks are on record but no longer in the firewall

Blocks deliberately do not survive a reboot — that is the anti-lockout net. The dashboard tells you, instead of letting you believe you are protected.

Active campaign against GLPI: 560 probes

Persistent and targeted — but you do not run GLPI, so it cannot touch you. It is background noise, useful only to know which wave you are in.

Zero open security vulnerabilities

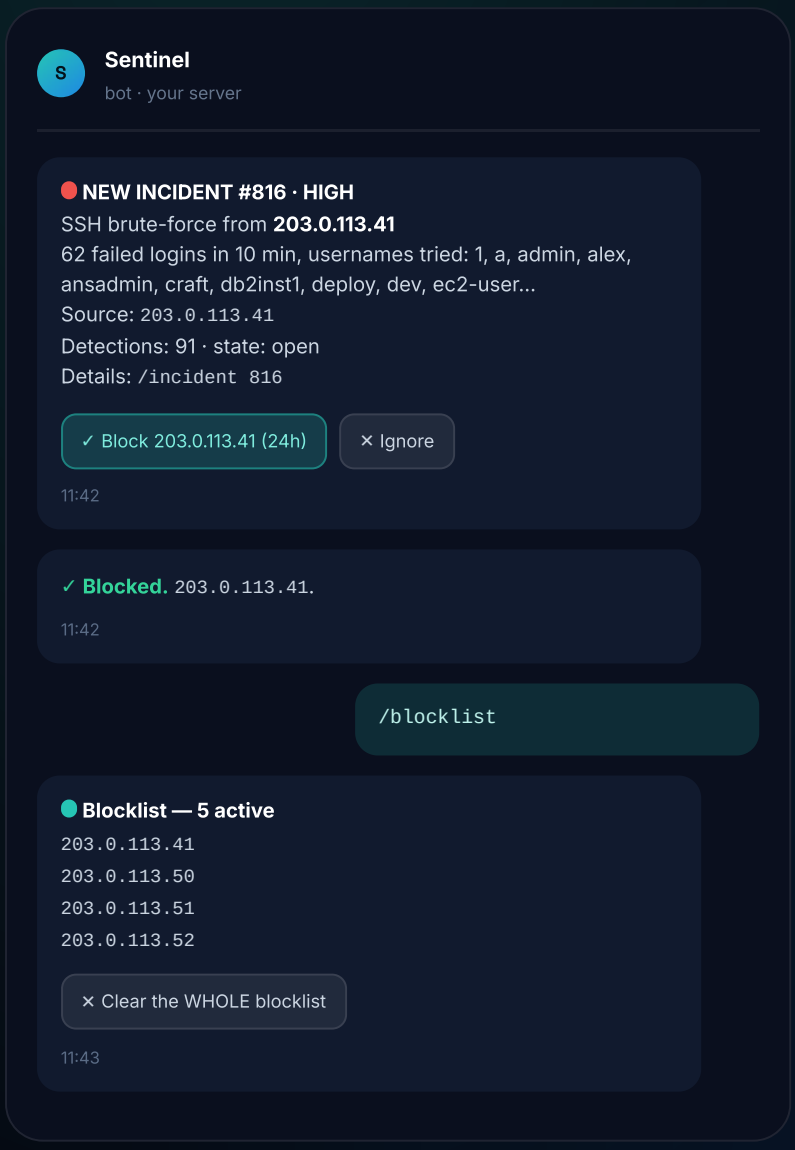
4 716 were fixed and confirmed by the following scan.

All 11 observations above are generated from the server’s real data, with no call to a model — the page renders instantly and says the same thing twice in a row.

THE EMERGENCY CHANNEL

The alert reaches your phone with the stop button under it.

Long-polling, so no extra open port and no public endpoint to forge. It works even if nginx or the certificate are broken — exactly when you need it.



The button is not the authority

Pressing it goes through the same role check as a typed command, and the executor refuses to block the admin address anyway — even if the button is pressed by mistake.

It survives the outage

The bot pulls messages rather than receiving them: no open port points at it and there is no public endpoint to forge. If nginx, TLS or the dashboard go down, the channel stays up.

Read is green, change is red

Commands that change something ask for a separate confirmation, and the destructive ones ask for a second one that restates the target.

A real transcript: the messages, the button labels and the command names are the ones the bot sends, not a mock-up.

FROM PACKET TO DECISION

What happens between the attack and the button on your phone.

Five stages, each with a job it does not hand to the next one. The deterministic part decides; the model explains afterwards.

01 · COLLECTION

Six sources, filtered at the collector

sshd, sudo/su, nginx, auditd, Suricata. Noise is rejected before the database — 96 000 engine diagnostics a day never get there.

02 · DETECTION

Deterministic rules, every 10 seconds

Honest thresholds over indexed SQL: brute-force, web enumeration, IDS alerts, volume anomaly. Zero cost per event.

03 · INCIDENT

Deduplicated by fingerprint

400 attempts from one address become one incident, not 400. The actor gets a country and an operator from local geo databases.

04 · AI TRIAGE

Judgement, not scores

The model recalibrates severity and writes the summary. The deterministic verdict stays next to it — disagreement is visible.

05 · DECISION

Yours, with one tap

Auto-block exists and is tested, but ships **off**. The first 72 h show you what it *would have* blocked.

× The model does not decide blocks

Detection, scoring and the decision are deterministic and run at zero cost. An unavailable API or an exhausted budget changes nothing about what gets blocked.

× The model executes nothing

The plan it writes goes through a deterministic validator that refuses it if it is not safe. The model drafts, the validator decides.

× The model is not on the critical path

API down or budget exhausted: detection, blocking and alerting carry on, marked “AI analysis unavailable”.

× The model does not exceed the budget

The cap is checked before every call, and the cost per incident is visible in the dashboard.

Detection runs at zero cost per event; the model is called only for incidents that pass the severity threshold — typically a few dozen a day.

Scanning is the easy part. What you fix first is the hard one.

Prioritisation from **CVSS × EPSS × KEV × exposure × criticality**. A CVSS 6.5 on the CISA actively-exploited list beats a 9.8 that nobody is exploiting.

sentinel.example.com/patches/3 — **Patch plan #3** — VALIDATED

Risk level	Impact	Estimated downtime	Reversible	Needs reboot
high	single-service	0s	yes	no

TARGET · HASH

TARGET `postgresql · hash 909a1d35a818de31…`

PREFLIGHT — CHECKS BEFORE

check_disk	no_incident	curl_installed
disk_free	no_open_incident	pkg_version

APPLY · ROLLBACK

APPLY `dnf -y update curl`

ROLLBACK `dnf -y downgrade curl-7.76.1-29.el9`

BACKUP `rpm_state · curl`

× Exploited beats theoretical

A vulnerability on the CISA actively-exploited list rises above a higher score that nobody is using in the wild.

× Exposure beats inventory

What is reachable from the internet is patched before what sits behind the firewall with the same CVE.

× Critical assets are never automatic

Databases and anything you marked critical never get an automatically generated patch — they get a plan you approve.

A real plan, generated automatically and validated on the first attempt. The rollback pins the old version correctly — the boring solution that works.

HOW A PATCH IS BUILT

Six steps, and the model is only present at the first one.

01 **Generation**

From the deterministic facts of the scan. The model is given no shell.

02 **Validation**

Every command is a list, the first element from an allowed list, `rm` forbidden. Invalid → one retry → rejected.

03 **Approval**

Two confirmations on Telegram, a single-use token tied to the hash.

04 **Backup**

A point verified by re-reading the archives. Without it, the patch does not start.

05 **Application**

Step by step, written to the database *before* each command.

06 **Rollback**

Automatic on any failure. A failed rollback is a separate, more serious state.

> The model is present at step 01 and nowhere else. Everything after it is deterministic: a validator that refuses, a single-use token, a backup that is re-read before it counts, and a rollback that runs on any failure.

× **The model is given no shell**

It writes a plan from the deterministic facts of the scan. It never runs a command, on any host.

× **One retry, then a refusal**

An invalid plan gets a single regeneration. The second failure is a refusal, not a workaround.

× **Written before it runs**

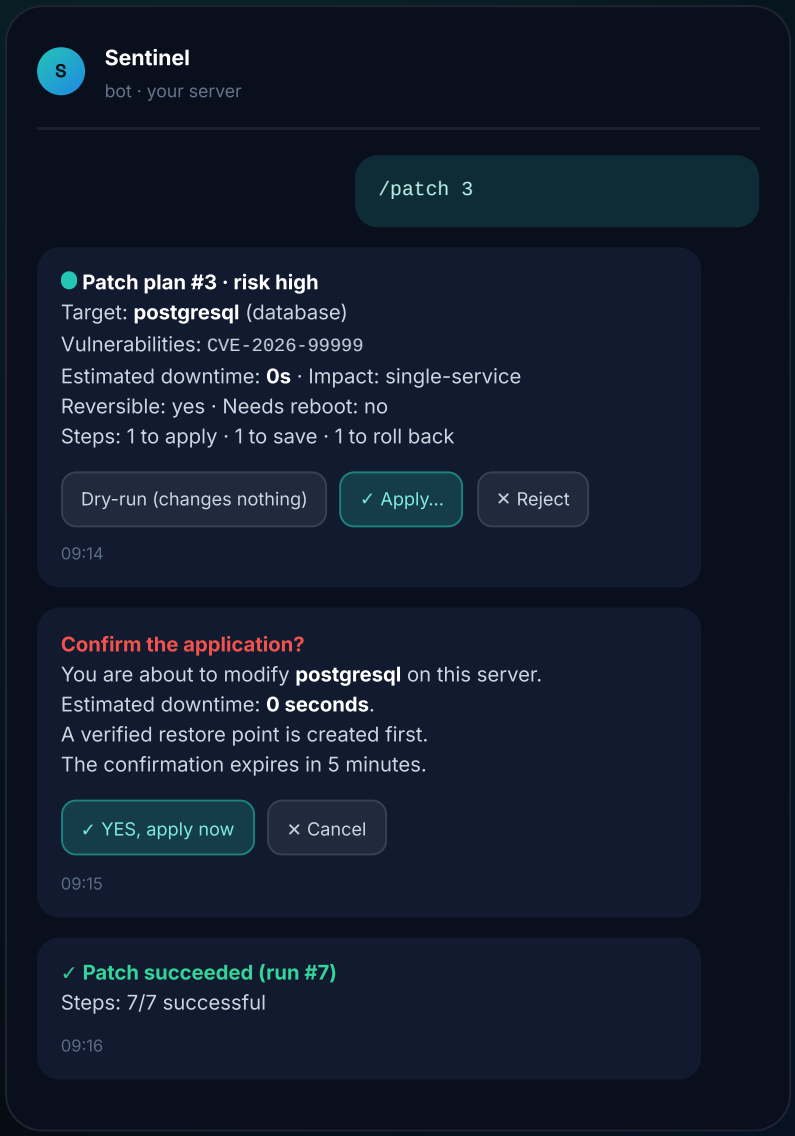
Each step lands in the database before the command is executed, so an interrupted run is readable afterwards.

Patches are generated automatically; they are never applied without two explicit confirmations.

PATCH APPROVAL

The first tap applies nothing.

A button lives in a chat history forever. So the button *is not* the authority — it carries a reference to one: a single-use token, tied to the chat and to the plan hash, with an expiry.



The second screen adds information

Otherwise "are you sure?" is just a dialog people close by reflex. It restates the target, the downtime and what is created before anything is touched.

A regenerated plan kills old buttons

If the plan is regenerated in the meantime, the hash changes and every previous button dies.

The token is single-use

Tied to the chat and to the plan hash, with an expiry. Forwarding the message forwards nothing usable.

Generating the plan is automatic. Applying it takes two explicit confirmations, from the phone, by someone with the right role.

SAFETY ARCHITECTURE

An agent that can block the internet must be judged by what it refuses.

Everything privileged goes through a single root process of a few hundred lines, with no dependencies and no imports from the rest of the application. The rest runs unprivileged.

× It cannot lock you out

Two independent layers: the nftables table has `policy accept` with an allowlist ahead of the blocks, and the executor refuses the admin address outright. Verified live — the refusal works.

× The dashboard cannot block anyone

The web interface can only *cancel* a block, never create one. A compromised dashboard stays a data leak, not a takeover.

× `rm` is not on the list

Absent from the allowed binaries for patches. Deletion goes through a single operation, strictly limited to the backup directory, with the path re-checked after resolution.

× The restore script writes itself

Generated *inside* the executor, from what it sees on disk. Text coming from the unprivileged side never becomes an executable line run as root.

× An empty backup is not a backup

Archives are re-read and re-hashed at sealing. A zero-size artefact is rejected — otherwise it looks like a way back exactly until you need it.

× It cannot be fooled through logs

Log lines are wrapped as untrusted data, and the model can only return a structured verdict. A successful injection changes a label, never the system.

× Retention does not delete the last way back

The last restore point of an asset is never deleted, however old. A retention policy that can do that is data loss on a timer.

× Emergency flush

`touch /etc/sentinel/PANIC` clears the blocklist every 10 seconds while the file exists. Blocks do not survive a reboot — restarting is always a way out.

INSTALLATION

One command, and nothing changes until you say yes.

The wizard works out where it is running on its own: on your laptop it asks for a target and installs over SSH; on the server it skips the SSH half.

```
$ git clone https://github.com/dragosnimu/sentinel.git
$ cd sentinel && ./scripts/wizard.sh --dry-run
✓ AlmaLinux 9.8 (rhel family)
✓ python3.12 — 5208 MB memory — 94 GB free
✓ nginx owns 80/443 → “shared” mode is available
! port 8443 is taken by something else
--dry-run: nothing was changed.
```

01 · ASKS

Nothing is guessed silently

The target, the domain, the nginx mode, your admin address, Telegram, the AI key. Where a sensible default exists, you see it before you accept it.

02 · CHECKS

Read-only

Distribution, memory, disk, who owns 80/443, whether the requested port is free, whether the domain resolves here. Nothing is touched.

03 · SUMMARISES

The last way out

Everything about to change, on screen, before the confirmation. An installer that has already edited nginx by the time it asks you is lying to you.

04 · INSTALLS

Numbered and resumable

Every step has a marker on disk. If one of your services stopped, the install rolls itself back.

× **Secrets never reach the process table**

The bot token and the API key are read with echo off and travel on stdin. As command-line arguments, `ps` would show them to every account on the host.

× **Two distribution families, one file**

RHEL (AlmaLinux, Rocky, CentOS Stream, Fedora) and Debian (Debian, Ubuntu). Everything that differs lives in `deploy/lib/distro.sh`.

× **What is unsupported is refused by name**

`unsupported distribution` — a host where the install failed visibly is recoverable. One that looks protected and is not, is not.

Answers can be saved and replayed unattended — a second server, or a rebuild, asks no questions at all.

STATUS

What is delivered and running in production.

514	31	7	13	0
automated tests	safety invariants	systemd services	schema migrations	CDN dependencies

Collection from six sources, deterministic detection, incidents with AI triage, blocking with a button on Telegram, vulnerability scanning with KEV prioritisation, patch generation and application with a verified backup and automatic rollback. A web interface with no third-party JavaScript, a strict security policy and zero external resources.

Auto-block ships **off** — the machinery is complete and tested, but the first 72 hours are for observation, so that you find the false positives before you cut someone off at 3am. Patches are generated automatically; they are never applied without two explicit confirmations.

× What is not covered yet

Container logs are not collected, there is no dedicated file-integrity monitor beyond auditd, and the Debian install is covered by tests but not yet verified on a real Debian host.

× How you get it

Install, Managed or Fleet — the licence is per server, and the initial configuration plus the 72-hour calibration are included in every tier.

NEXT STEP

Thirty minutes, free, with no obligation.

You tell us what server you have and what runs on it. We tell you what is exposed, what is attacking you right now and whether Sentinel makes sense for you — including when the answer is “not yet”.

cryptoitdata.eu/contact
book a consultation

cryptoitdata.eu/sentinel/en
the product page