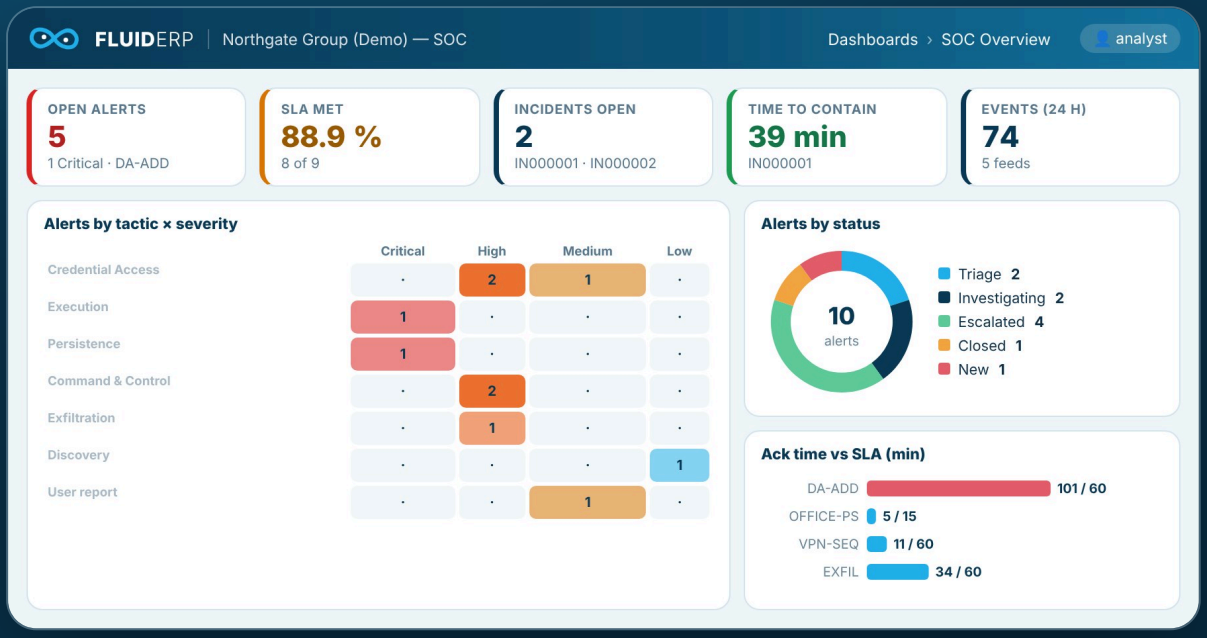
FLUID ERP · USER MANUAL

FLUID Security

Detect. Respond. Prove.

Event capture in the SOC's own wire formats, detection rules as data, the alert queue, incidents, approved response actions, threat intelligence, vulnerabilities, controls and the signed, sealed record.

CEF · JSON Lines · syslog · STIX | Detection rules as data | Alert queue & SLA | Response actions | Signed & sealed incidents | Vulnerabilities & Essential Eight



Contents

1 About this manual 3

Who it is for 3

Conventions 3

2 How FLUID works: capture once, post everywhere 4

Three kinds of action 4

Rules you can rely on 5

3 Getting started 6

Signing in 6

The launcher 6

Anatomy of a screen 7

Everyday actions 8

4 Capturing events 9

Load events from a feed or file 9

Send the same file again safely 10

Understand enrichment 11

5 Detection rules and alerts 12

Write, test and sign a rule 12

Run detections 13

Run it again — nothing new 14

Triage the alert queue 15

6 Incidents and response 16

Escalate an alert 16

Disable an account, block an address 17

Isolate a host and contain 18

Close, sign and seal 19

7 Exposure: vulnerabilities, controls and staff reports 20

Manage vulnerabilities 20

Track controls 21

Report something suspicious (security app) 22

8 Critical dashboards 23

SOC overview 23

ATT&CK coverage 24

Exposure and risk 25

9 Reports 26

Running and printing a report 26

Report catalogue 26
Incident register 27
Rule quality 28
Vulnerabilities past their date 29
Control posture 30
10 Matching 31
How every matching screen works 31
Indicators against events 32
Scanner hosts against the asset register 33
11 Messages, refusals and what to do 34
12 Data reference and glossary 35
Tables you will meet 35
Glossary 35
A Walk it once on the demo package 37
For the administrator 37

1 About this manual

This manual explains how to run a security operations centre on FLUID Security: loading events, writing and signing detection rules, triaging alerts, escalating incidents, recording approved response actions, closing and sealing incidents, managing vulnerabilities and controls, and reading the SOC dashboards and reports. The worked examples follow the Northgate Group SOC at 07:30 on 13 October 2026.

Who it is for

READER	READ THESE CHAPTERS
Analysts	3, 4, 5, 6, 10
Responders	3, 6, 7
SOC lead	3, 5 (rules and signatures), 6, 8, 9
CISO	6 (close and seal), 8, 9
GRC	7, 8, 9, 10
System owners (security app)	7 (My Findings, Report Something Suspicious)

Conventions

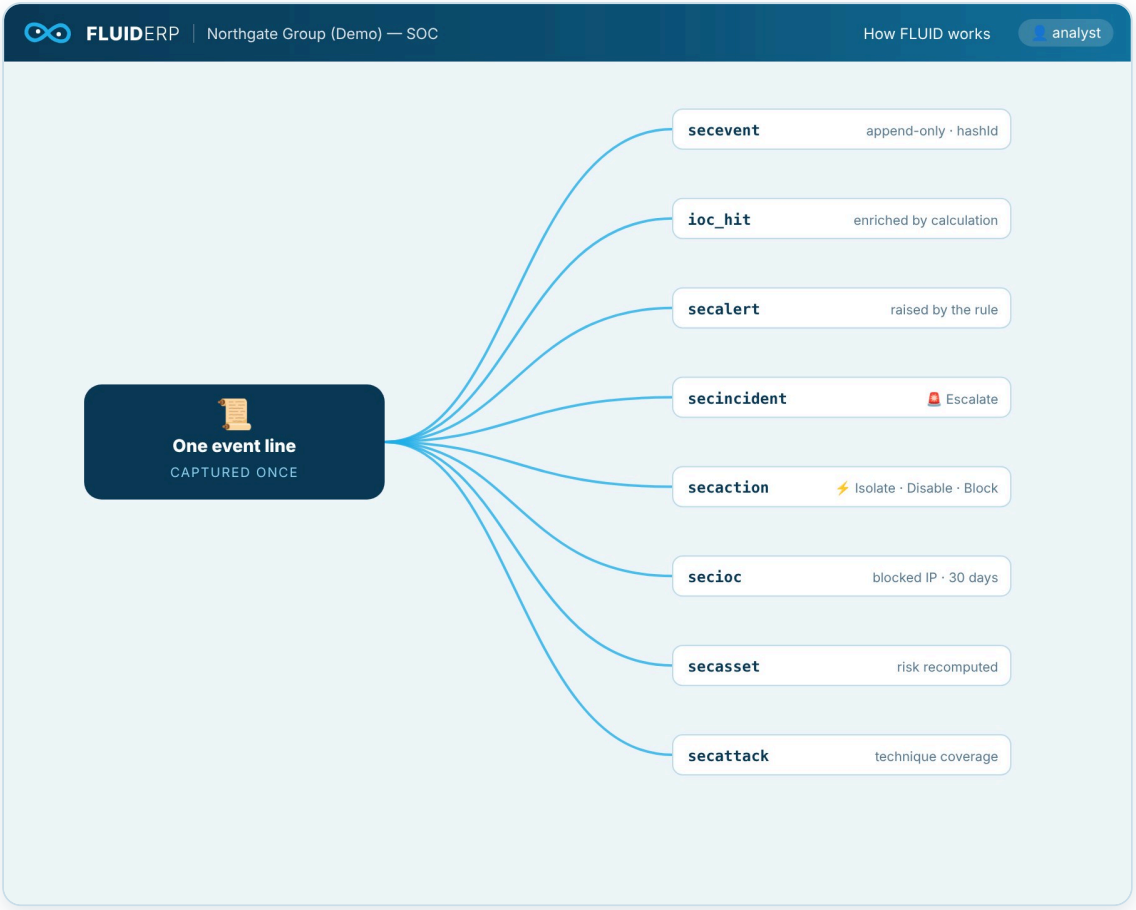
YOU SEE	MEANS
Bold names such as **Shifts**	A tile on the launcher, a screen, a tab or a field label, exactly as it appears in FLUID.
A button with an emoji, e.g. 🚚 **Post the Shift**	A posting button. Posting buttons are aqua; they write to other tables. 💾 **Save** is navy and saves only the record in front of you.
A field marked ƒ	A calculated field. You never type it — the calculation engine fills it from other fields or other tables, and keeps it right when they change.
`mneshift`	The name of the underlying table (collection). Useful when talking to your administrator or reading a report definition.
SH000006, MO000001 …	Record numbers. They are issued automatically by counters and carried on every row a button writes, so you can trace any figure back to the record that caused it.

ABOUT THE EXAMPLES

The assets, events, rules, alerts, incidents and seals follow the Stage 71 Northgate story (nine episodes, IN000001–IN000003). Alert ids, some vulnerability and control rows, and dashboard histories are illustrative. Screenshots come from the FLUID Security demonstration film.

2 How FLUID works: capture once, post everywhere

An event is captured once, as a row. The rules raise alerts from those rows; escalation makes the incident from the alert; the playbook writes the approved action and the indicator; the CISO's signature seals the record. Nothing is copied from the SIEM into a ticket or a spreadsheet.



One event line fans out to enrichment, the alert, the incident, response actions, indicators, asset risk and ATT&CK coverage.

Three kinds of action

ACTION	WHAT HAPPENS	TYPICAL BUTTON
Capture	A person (or a feed, a file, a phone app) records something that happened — once. It is saved on its own table and gets a number.	💾 Save · a form on the phone · a file import or feed
Post	A posting button reads the captured record and writes every row that follows from it — ledgers, stock, other documents, stamps — each carrying the source number. Most posting buttons are one-way: the record is marked posted and cannot be posted again.	🚨 Escalate · ⚡ Block IP · ✍ Close (CISO signature)

ACTION	WHAT HAPPENS	TYPICAL BUTTON
Derive	Calculations keep totals, balances, KPIs, flags and due dates right. Dashboards and reports read those rows directly; nothing is copied into a spreadsheet.	none — fields marked ƒ, dashboards, reports

Rules you can rely on

1. **Nothing is typed twice.** If a number exists in one record, every other table that needs it receives it from the button, not from a person.
2. **A posted record is sealed.** Pressing a posting button a second time writes nothing and says so. To change a posted figure you raise a correcting document (a credit note, a reversal, a correction) — which is itself captured once and posted.
3. **Every row is traceable.** Each row a button writes carries the reference of the record that caused it (for example `ref = IN000002`). Drill from any dashboard figure to the rows behind it and from there to the source record.
4. **Rules are enforced at the server.** Approvals, segregation of duties, embargoes, seals and signatures are checked by the gateway for every user, not just on screen.
5. **Imports are idempotent.** Files and feeds are keyed so that sending the same data again updates rows instead of duplicating them.

> **READ A POSTING PANEL**
>
> When you press a posting button, a panel lists every table it is writing to, one tick at a time, and ends with a green line such as “Posted — 5 writes to 5 tables from one capture”. That list is exactly what the button changed; the receiving tables show the same reference number.

3 Getting started

Signing in

Open the FLUID app (Windows, Android) or the browser client and sign in with the user name and password your administrator gave you. Each sign-in belongs to a group; the group decides which tiles you see and which fields and buttons you may use.

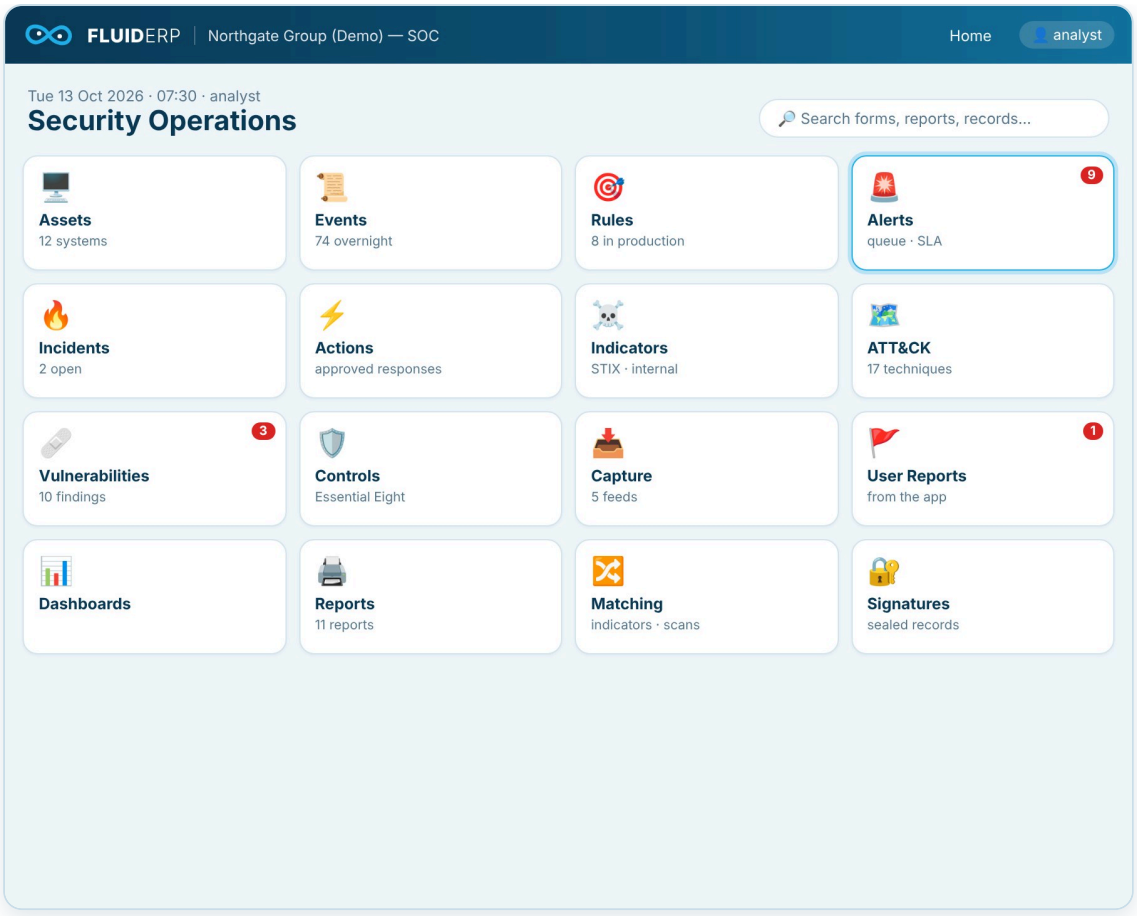
SIGN-IN	GROUP / DESK	USES FLUID TO ...
analyst	Analyst	Events, alert queue, triage, enrichment, Add to Intel
responder	Responder	Incidents, response actions
soc_lead	SOC lead	Rules (sign to Production), approve actions
ciso	CISO	Close incidents with a signature, accept risk
grc	Governance, risk & compliance	Vulnerabilities, controls, posture reports
fiona.lowe@ / raj.patel@ / dale.ferris@northgate.example.org	System owners (security app)	Report Something Suspicious, My Systems, My Findings

DEMO SIGN-INS

In the demo package staff passwords are the same as the user name; the security-app owners use owner123. Change every password at go-live.

The launcher

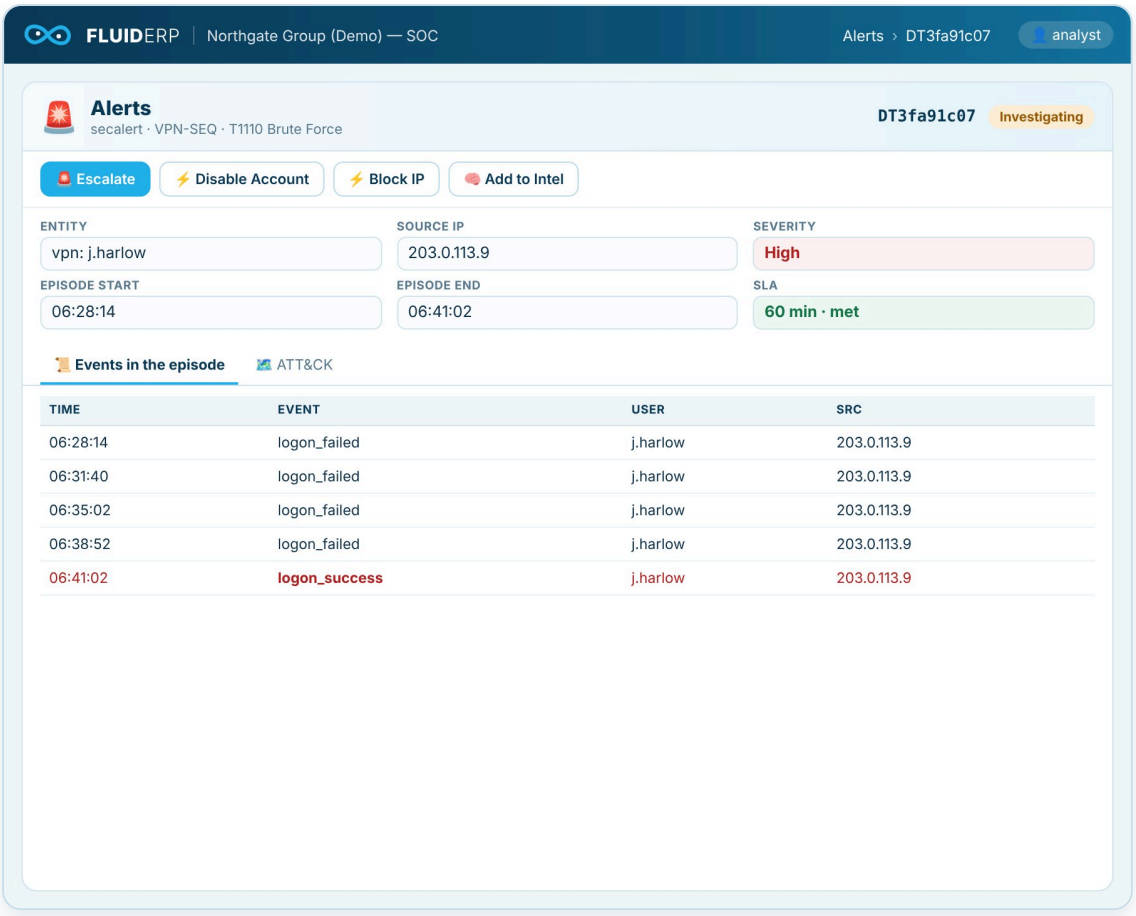
After sign-in you see the launcher: a grid of tiles for the screens, dashboards, reports and matching screens your group may open. A red badge on a tile counts records waiting for you. Use the search box to jump straight to a form, a report or a record number.



The SOC launcher: nine alerts in the queue, three vulnerabilities past their date, one user report.

Anatomy of a screen

1. **Header** — the form name, the table it belongs to, the record number and a coloured **status** pill (for example Closed, Posted · sealed, Approved).
2. **Toolbar** — 💾 **Save** (navy) saves this record. Aqua buttons are **posting buttons**: they write to other tables. Outline buttons recalculate (🧮), print (🖨️) or open related screens.
3. **Fields** — typed fields are white; calculated fields (ƒ) are tinted and read-only. Red, amber and green fields are flags raised by a calculation (expired, due soon, OK).
4. **Tabs** — the record's lines and related records (lines, movements, attachments, signatures). Child rows are saved with the parent.
5. **Banners** — an amber or red banner explains why a record is locked, sealed, embargoed or refused. Read it before calling support; it names the rule.



An alert: entity, severity, the episode's start and end, the SLA, the response buttons and the events in the episode.

Everyday actions

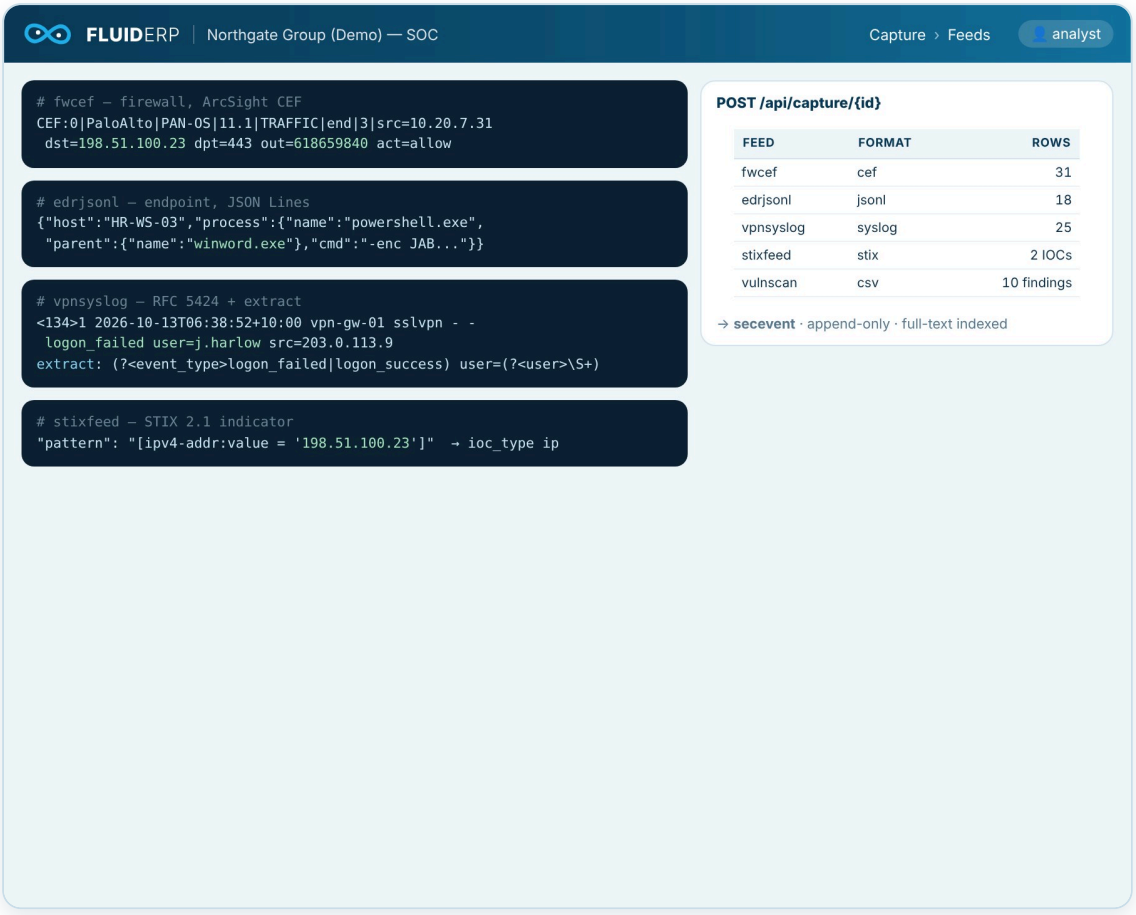
TO ...	DO THIS
Create a record	Open the tile, press ➕ **New**, fill the white fields, press 💾 **Save**. The number is issued on first save.
Find a record	Use the search box on the launcher or the filter row above any list. Record numbers, names and (where enabled) full text are searchable.
Refresh calculated fields	They refresh on save. 🧮 **Recalculate** re-runs them on demand (for example after a related record changed).
Print or save as PDF	🖨️ **Print** opens the printed copy in the document viewer; choose 🖨️ **Print / PDF** for paper or a PDF, or 💾 **Save** for the file.
Work offline (phone apps)	Forms marked as captures keep working with no signal; they wait in the **outbox** and send when you are back in range.

4 Capturing events

Events arrive in the formats your tools already speak. A capture definition maps them into one shape of row on secevent, which is append-only.

Load events from a feed or file

Who **administrator / scheduled task** | Where **Capture › Feeds · POST /api/capture/{id}** | Table **secevent**



The four wire formats and what they became: firewall CEF, endpoint JSON Lines, VPN syslog with extract, STIX indicators.

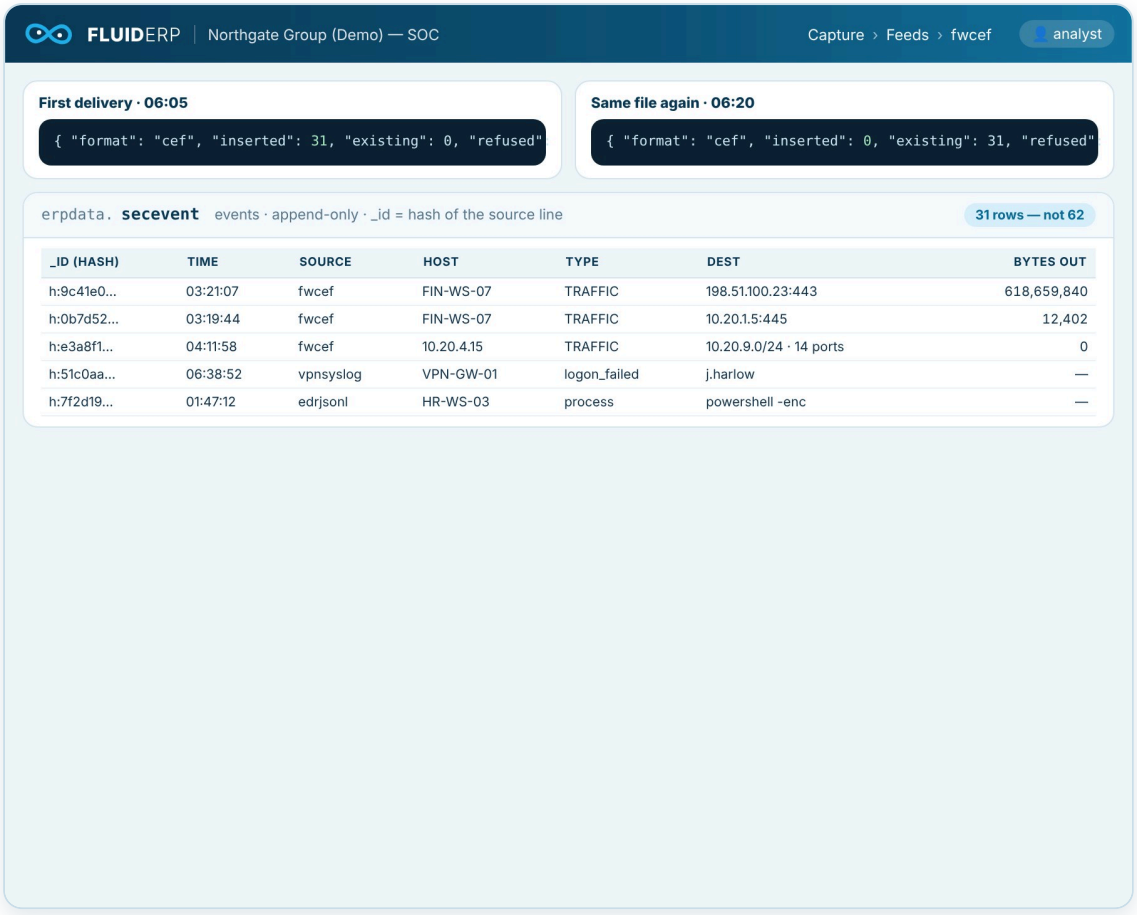
1. Send the file or stream to the capture endpoint for its definition: `fwcef` (CEF), `edrjsonl` (JSON Lines), `vpnsyslog` (syslog), `stixfeed` (STIX 2.1), `vulnscan` (CSV).
2. The definition parses the header and fields (CEF extension, flattened JSON names, RFC 3164/5424 syslog, STIX patterns) and, with **extract**, pulls named fields from free text by regular expression.
3. The reply lists inserted, existing and refused rows, with the reason for every refusal.

> **TIP**
> Events are full-text indexed. Search them from the Events screen like any other record.

Send the same file again safely

Who **administrator** | Where **Capture › Feeds**

With **hashId** a row's id is a hash of its source line. Re-sending a file after a link drop inserts nothing it already has and never overwrites what an analyst has touched.

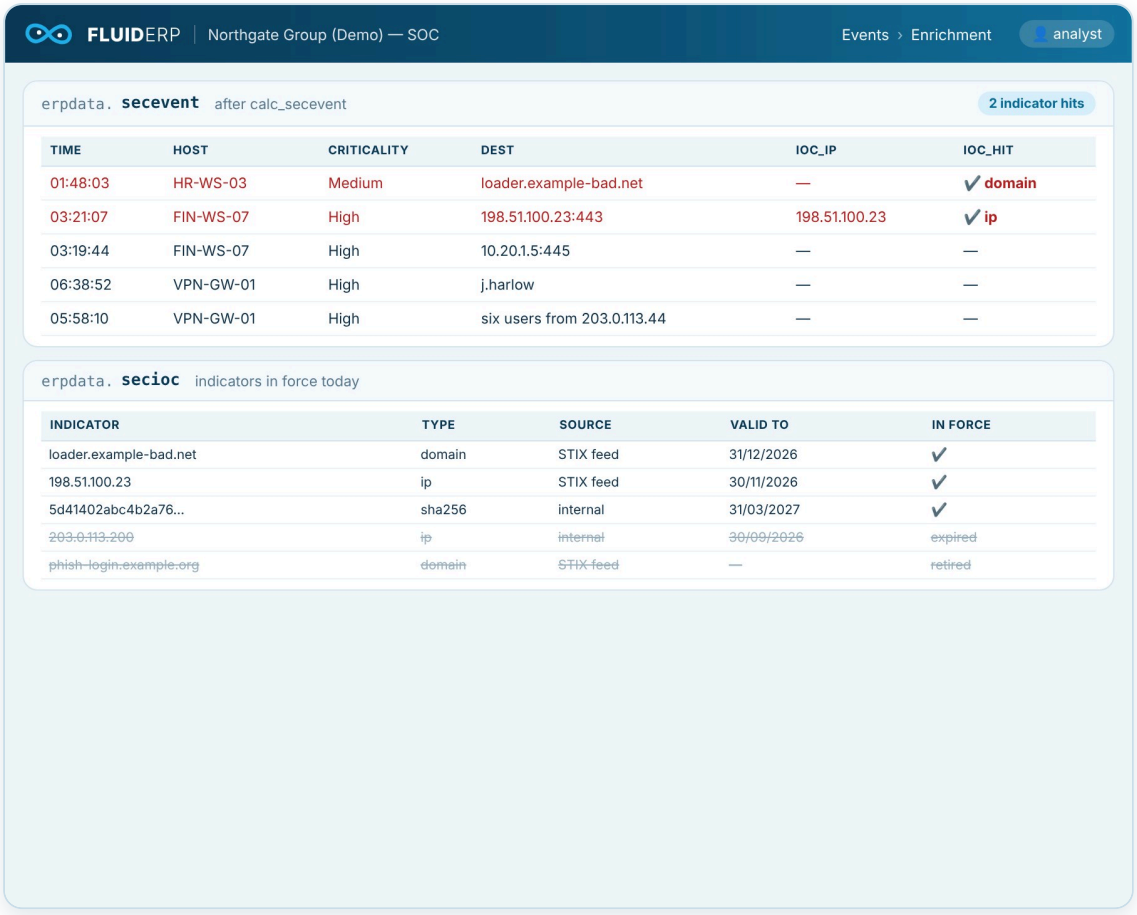


The night's firewall file sent twice: 31 rows, not 62.

Understand enrichment

Who **analyst** | Where **Events** | Table **secevent · secioc**

Every five minutes the calculation compares each event with the indicators in force on its date. Expired and retired indicators do not match.



calc_secevent stamps ioc_hit, ioc_ip / ioc_domain / ioc_hash and the asset's criticality.

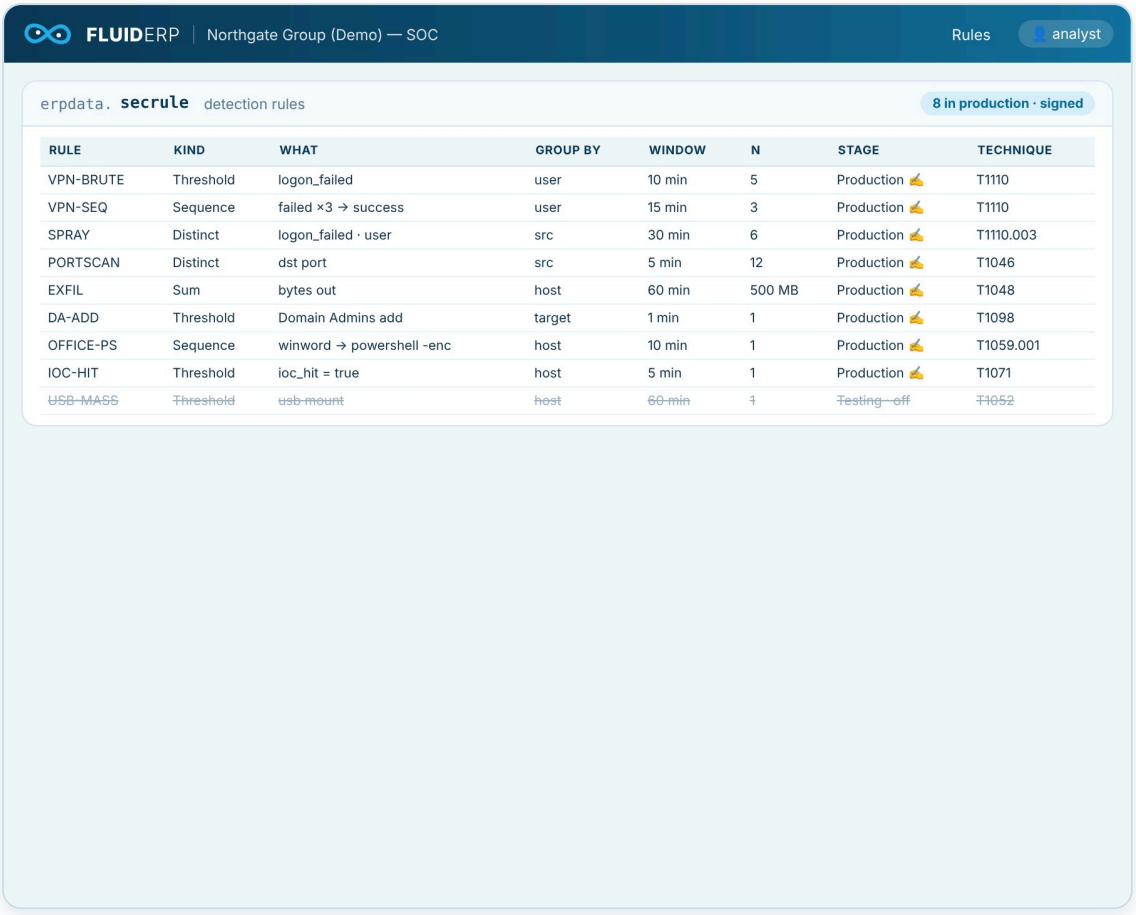
> **NOTE**
> Enrichment happens at the next calculation, not at capture. A rule that reads ioc_hit sees a new event after the calculation has run.

5 Detection rules and alerts

A detection rule is a row: a filter, a group-by, a window and a number. The worker turns rules and events into alerts.

Write, test and sign a rule

Who **analyst (draft) · soc_lead (sign)** | Where **Rules** | Table **secrule**



Nine rules — eight in Production, signed by the SOC lead; the USB rule in Testing, off.

1. Create the rule: **kind** (Threshold, Distinct, Sum or Sequence), **event filter** (JSON with equality, $ne, $in, $gt, $contains …), **group by**, **window**, **threshold** (or distinct / sum field, or the sequence steps), severity, tactic and technique.
2. Stage **Draft → Testing**. Run the worker as a dry run to see what it would raise.
3. The SOC lead moves it to **Production** and signs with their password. The signature is bound to the kind, filter, steps, group-by, window, threshold and severity.

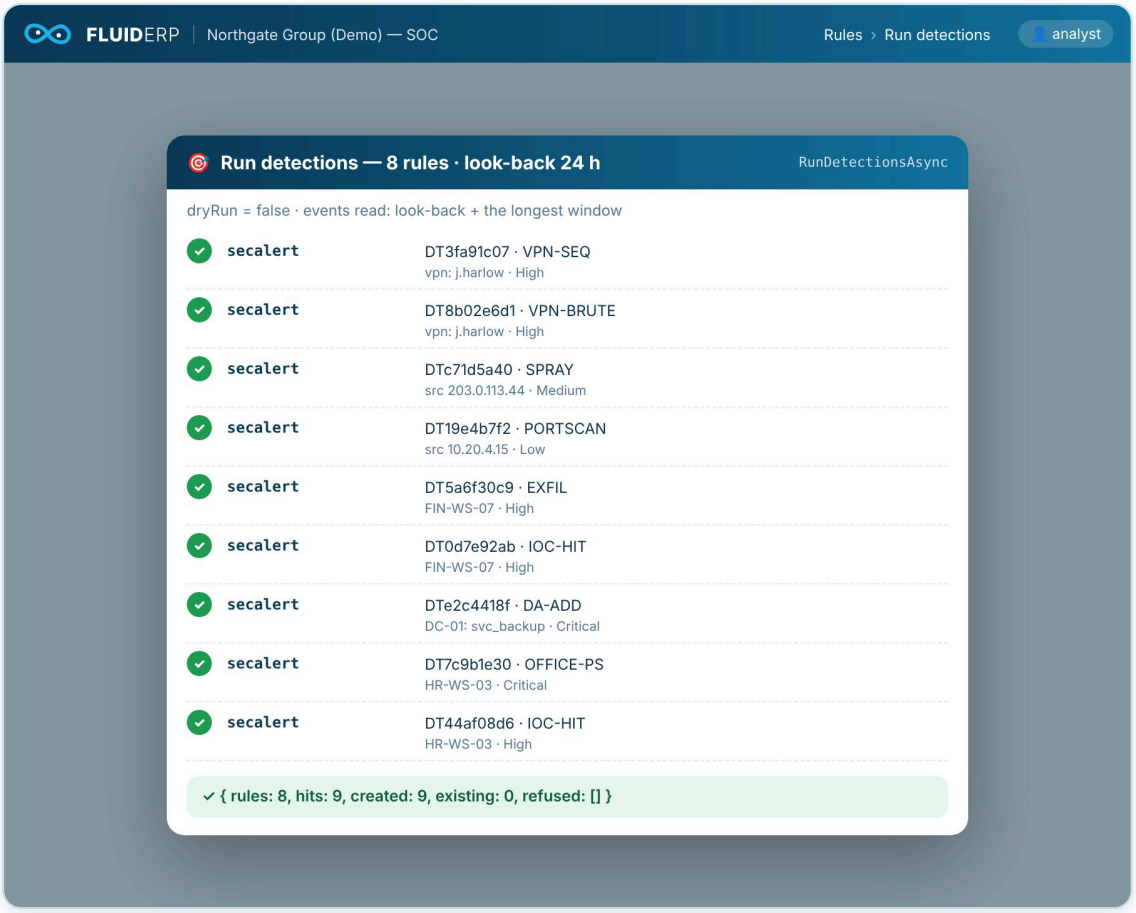
If the system refuses

YOU SEE	WHAT IT MEANS / WHAT TO DO
Rule did not parse — refused, with the reason	Fix the filter JSON or the steps. Other rules still run.

YOU SEE	WHAT IT MEANS / WHAT TO DO
Signature superseded — a reason and the lead's password are required	You changed a bound field of a Production rule. Tuning means a new signature.

Run detections

Who **administrator (scheduled every minute)** | Where **Rules › Run detections · POST /api/admin/detect/run**



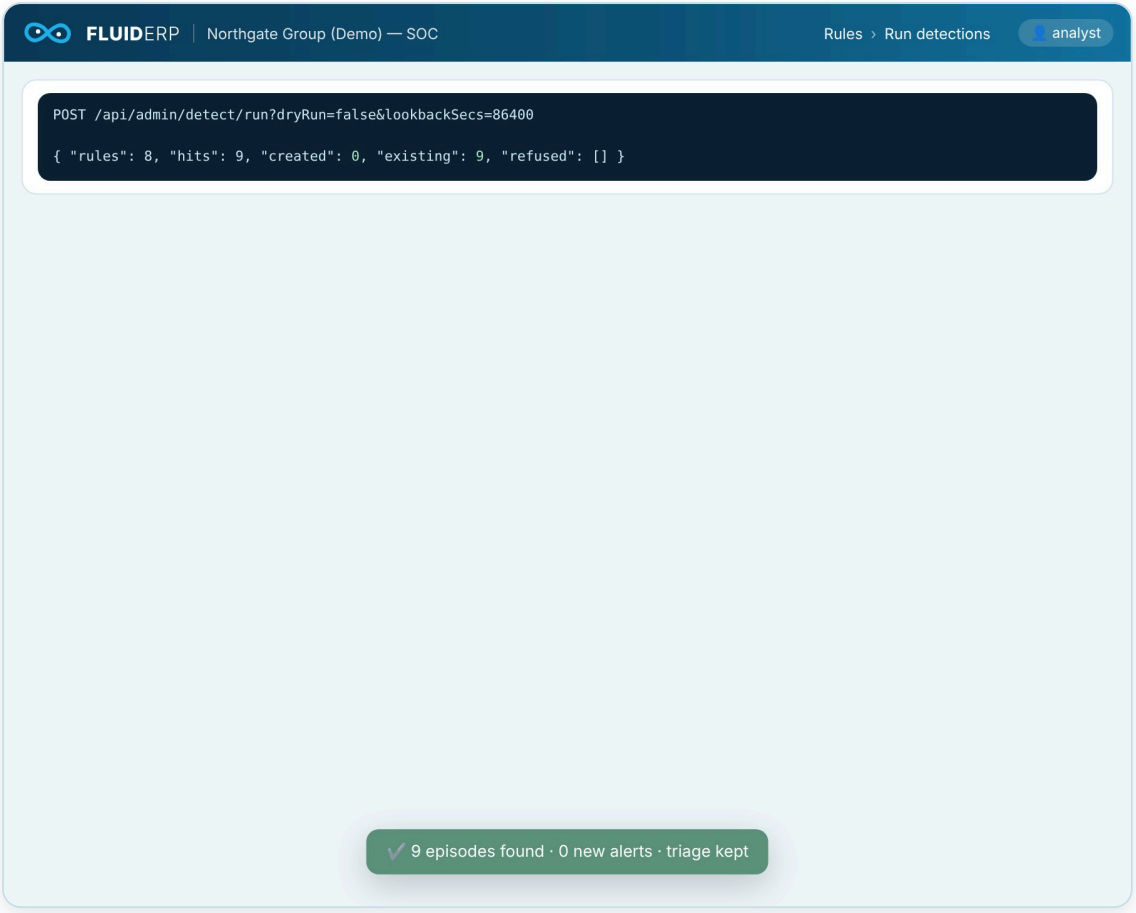
Eight rules over the night's events: nine episodes, nine alerts.

1. Schedule `POST /api/admin/detect/run?dryRun=false&lookbackSecs=86400` every minute with an administrator token.
2. The worker reads the events of the look-back plus the longest window, raises one alert per episode that started inside the look-back, and inserts only the absent ones.

What Run detections writes	RunDetectionsAsync
✓ **secalert**	one alert per episode; id = DT + hash of rule, group and first time; detected_at = when the episode ended

Run it again — nothing new

Who **administrator** | Where **Rules › Run detections**

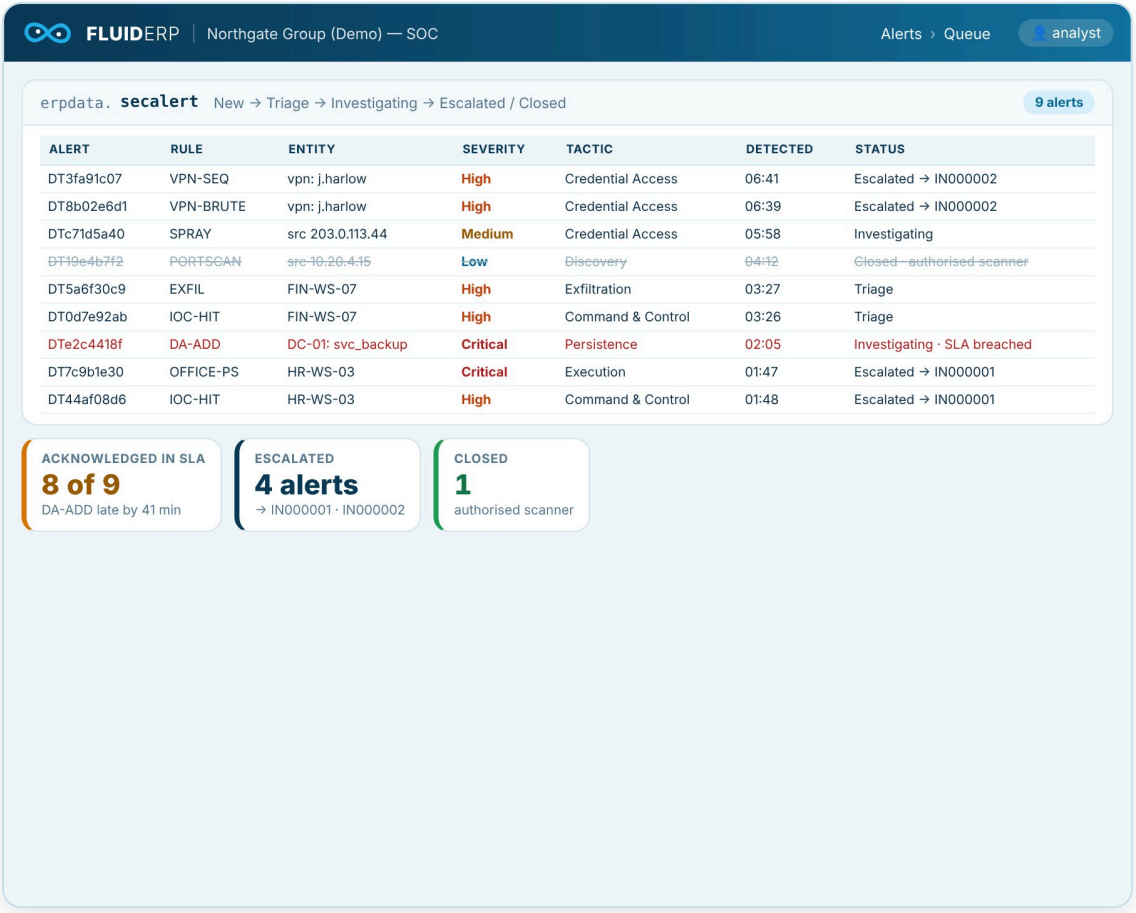


A second run: 9 hits, 0 created, 9 existing — triage untouched.

> **TIP**
> After an outage, increase the look-back to backfill. Alerts are id-stable, so a backfill raises only what was missed.

Triage the alert queue

Who **analyst** | Where **Alerts › Queue** | Table **secalert**



The queue with SLA status: the Domain Admins addition was acknowledged late; the port scan closed as the authorised scanner.

1. Work the queue from Critical down. Acknowledge each alert (**Triage**) inside its SLA: 15 / 60 / 240 / 1,440 minutes by severity, from when the episode ended.
2. Open the alert to see the events in the episode and the ATT&CK technique.
3. Move it to **Investigating**, **Escalated** or **Closed**. Closing requires a **disposition** (true positive, benign, authorised activity ...).

If the system refuses

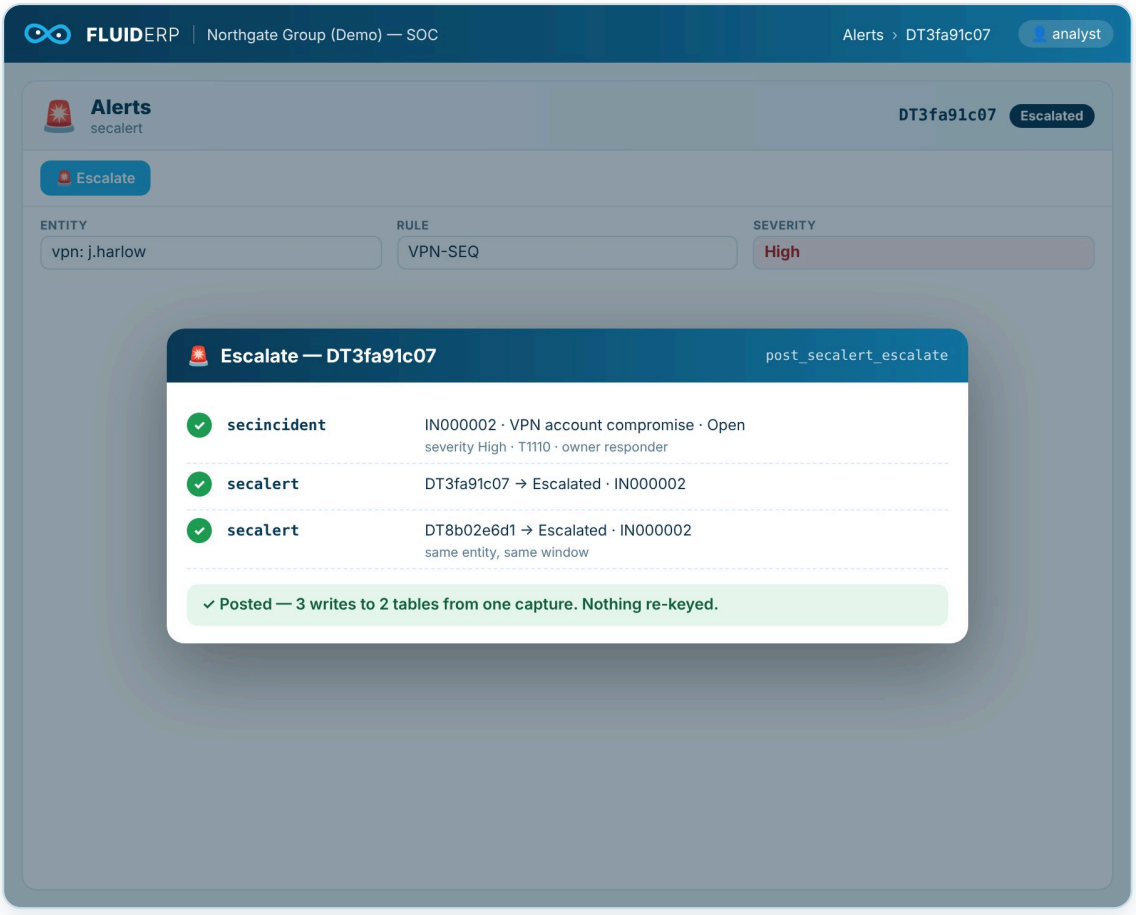
YOU SEE	WHAT IT MEANS / WHAT TO DO
A disposition is required to close	Choose why the alert is being closed.

6 Incidents and response

Escalation makes the incident from the alert. Responses are approved instructions written as rows; the change itself is made by a person or a connector that reads the row.

Escalate an alert

Who **analyst** | Where **Alerts › DT3fa91c07**



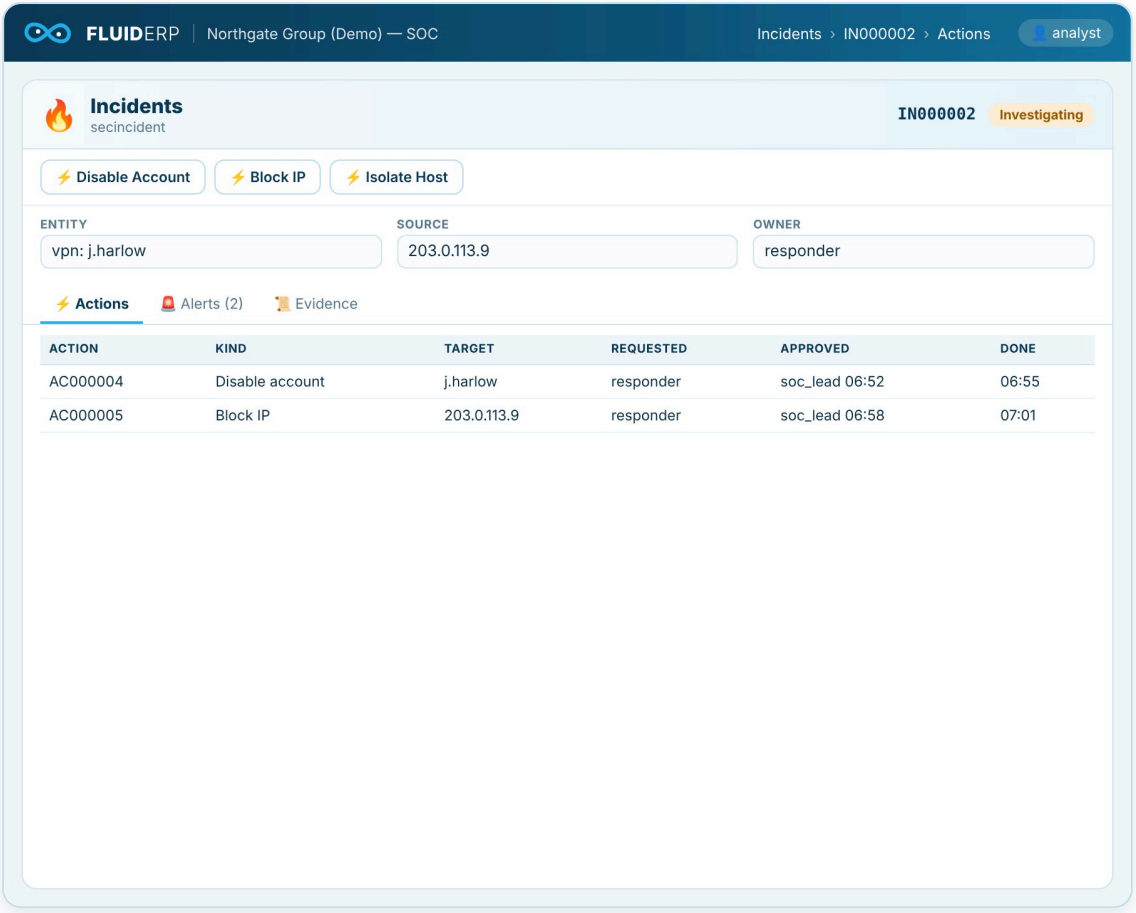
🚨 Escalate — IN000002 opened from the VPN sequence alert; the brute-force alert on the same account joins it.

1 Open the alert and press 🚨 **Escalate**.

What 🚨 Escalate writes	post_secalert_escalate
✓ **secincident**	IN000002 — entity, severity, technique and evidence from the alert; Open
✓ **secalert**	the alert → Escalated with the incident number

Disable an account, block an address

Who **responder (request) · soc_lead (approve)** | Where **Incidents › IN000002 › Actions** | Table **secaction · secioc**



AC000004 Disable account and AC000005 Block IP — requested, approved, done.

1. From the incident or the alert press ⚡ **Disable Account** or ⚡ **Block IP**.
2. The SOC lead approves the action; it can only be marked Done after it is Approved.
3. Whoever holds the firewall or directory makes the change and marks it Done.

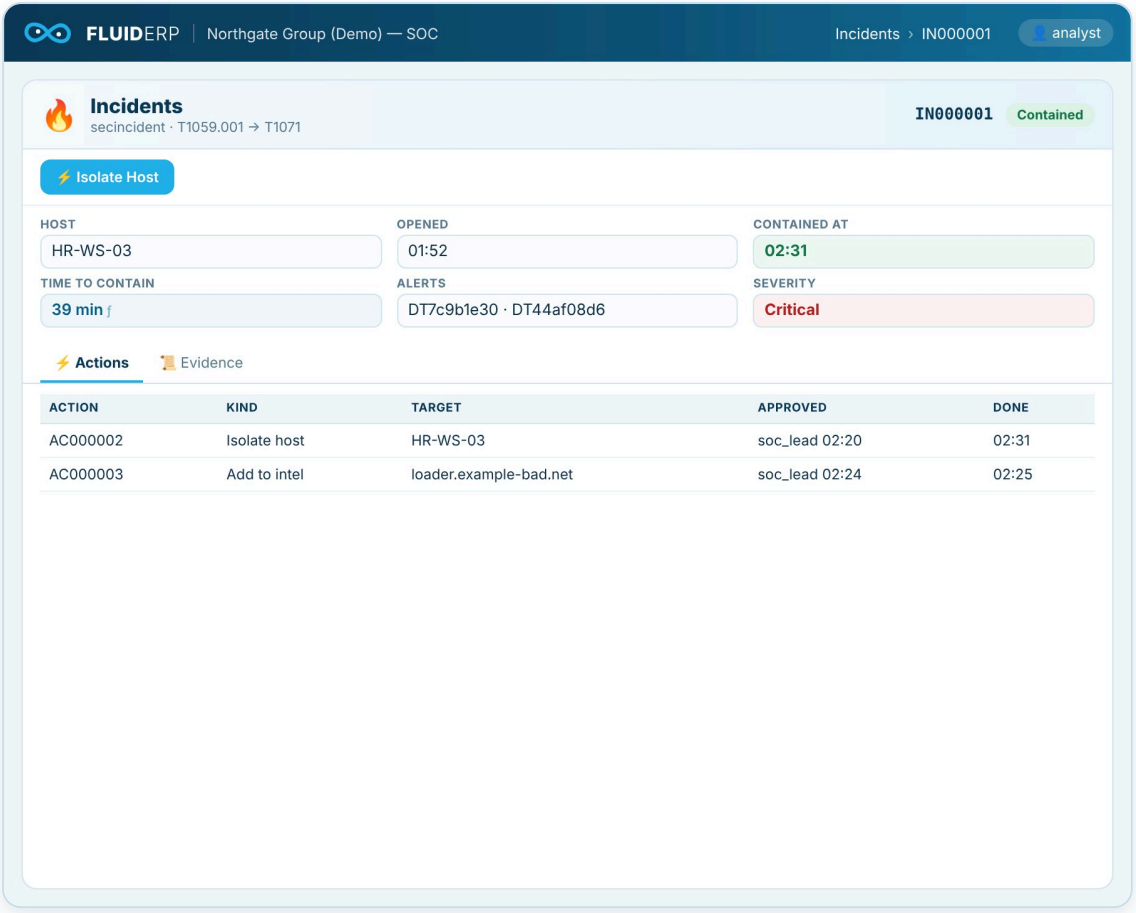
What ⚡ Block IP writes	post_secaction_block
✓ `secaction`	the action: who asked, who approved, when done
✓ `secioc`	the address as an internal indicator valid for 30 days — tomorrow's events are enriched with it

If the system refuses

YOU SEE	WHAT IT MEANS / WHAT TO DO
Disable Account refused — the alert has no user entity	Use Isolate Host or Block IP instead.
An action must be Approved before Done	Ask the SOC lead to approve.

Isolate a host and contain

Who **responder** | Where **Incidents › IN000001**

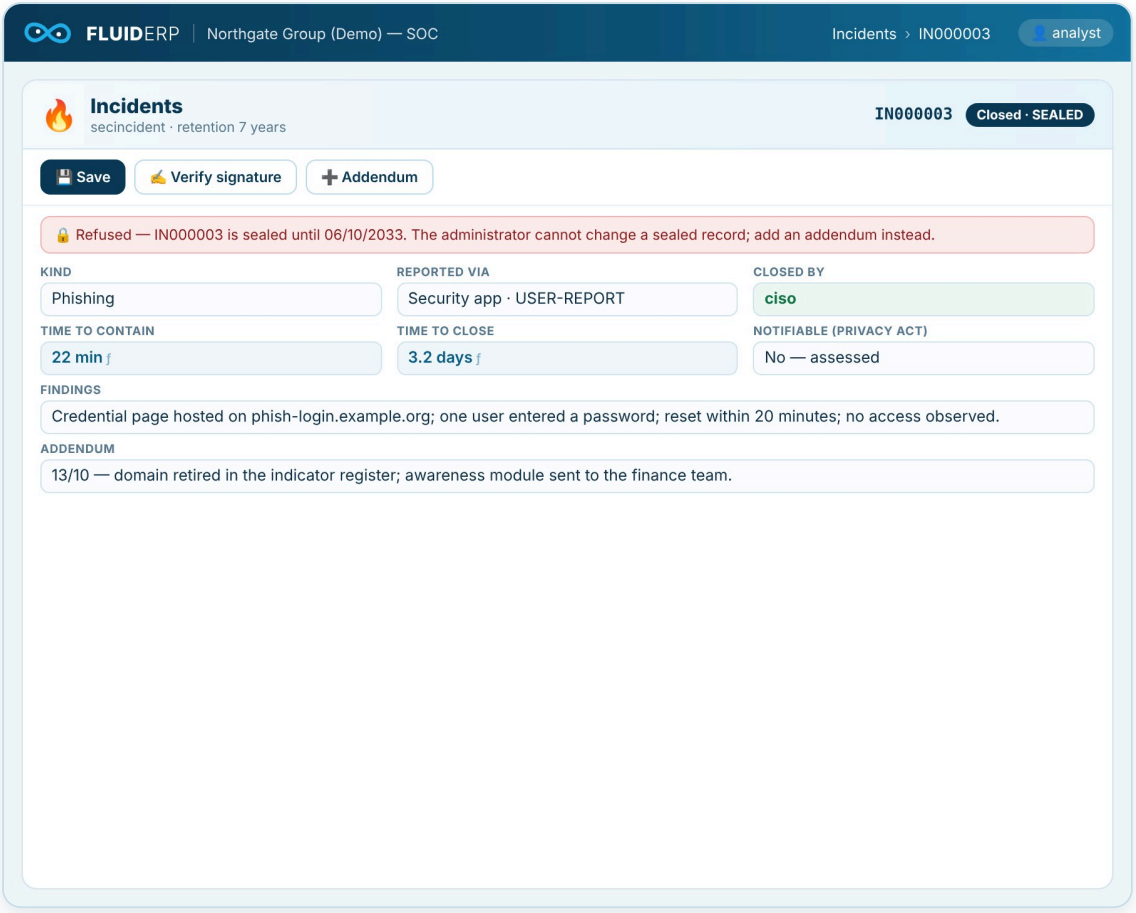


IN000001 contained at 02:31 — 39 minutes after it opened.

1. Press ⚡ **Isolate Host**; when the endpoint team confirms, enter **Contained at**.
2. Time to contain is calculated. Move the incident through Investigating → Contained → Eradicated → Recovered.

Close, sign and seal

Who **ciso** | Where **Incidents › IN000003**



IN000003 closed by the CISO with a signature and sealed for seven years; the administrator's edit is refused; an addendum is allowed.

1. Only the CISO can close an incident. Set **Closed**, re-enter the password and confirm the printed meaning of the signature.
2. The record is sealed: a fixity hash is written and every later change is refused — for the administrator too.
3. To add something afterwards use ➕ **Addendum**, which sits outside the signature.

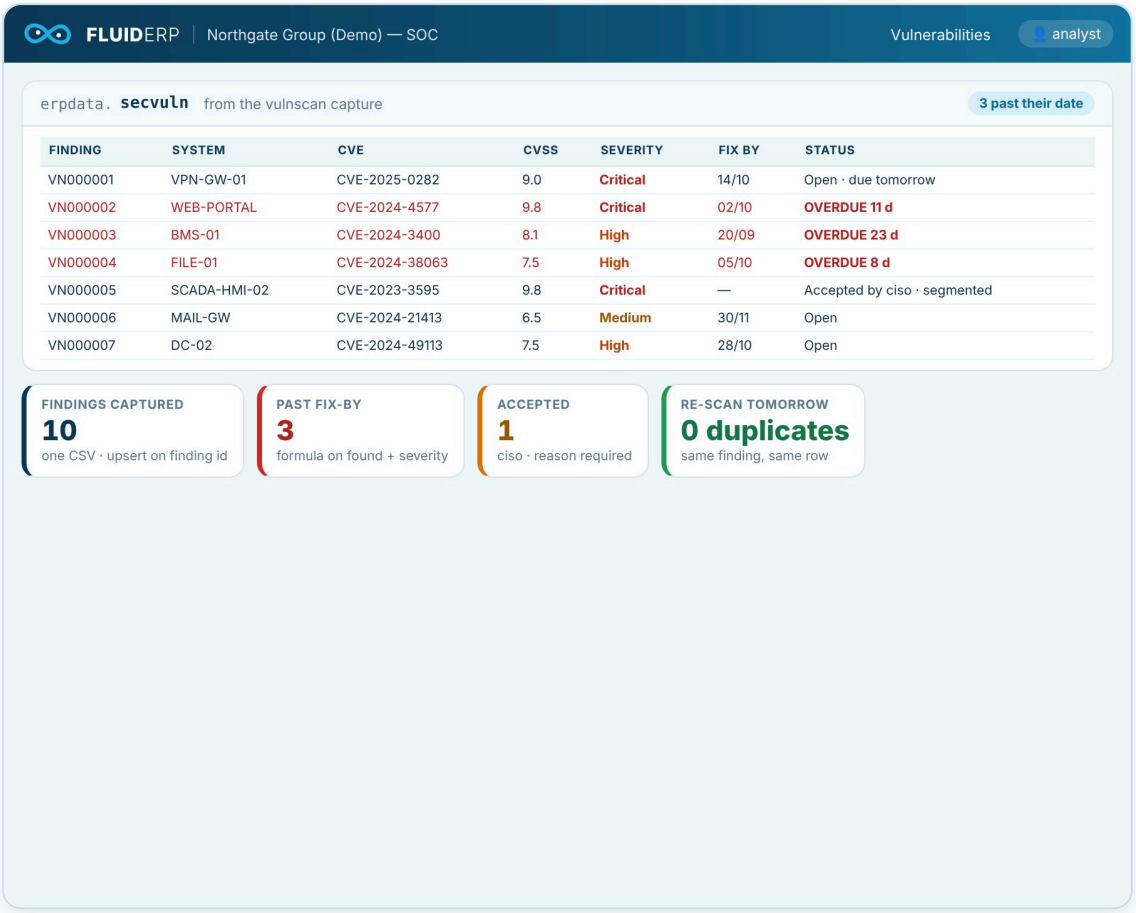
If the system refuses

YOU SEE	WHAT IT MEANS / WHAT TO DO
Refused — the record is sealed until …	Add an addendum instead of editing.
Only the CISO closes an incident	Ask the CISO.

7 Exposure: vulnerabilities, controls and staff reports

Manage vulnerabilities

Who **grc, system owners** | Where **Vulnerabilities** | Table **secvuln**



Findings from the scanner's CSV: severity from CVSS, fix-by from severity, overdue calculated.

1. Load the scanner file through `vulnscan`; the same finding re-scanned updates its row.
2. Severity follows CVSS; fix-by is 7 / 30 / 90 / 180 days by severity; overdue is a formula.
3. Owners see their findings in 🩹 **My Findings** on the security app. Accepting a risk is the CISO's decision and needs a reason.

Track controls

Who **grc** | Where **Controls › Essential Eight** | Table **seccontrol**

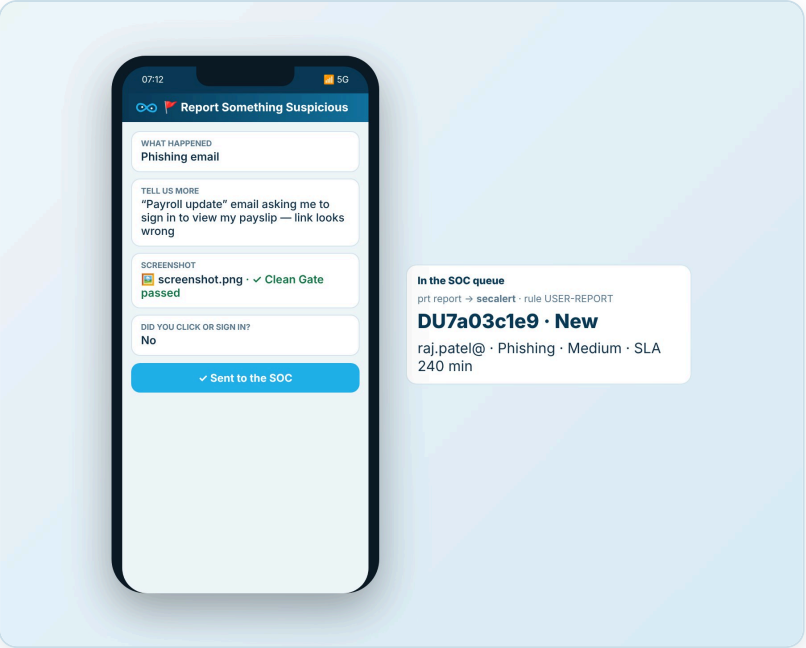


Essential Eight maturity now against target, with three tests overdue.

1. Record maturity, evidence, last tested and the test interval for each control (Essential Eight and ISO 27001 Annex A).
2. Next test and overdue are calculated.

Report something suspicious (security app)

Who **any staff member** | Where **Security app › 🚩 Report Something Suspicious**



A phishing report with a screenshot, checked by the Clean Gate, arrives as a New alert (rule USER-REPORT).

1. Choose what happened (phishing email, lost device, a login you did not make).
2. Describe it and attach a screenshot; say whether you clicked or signed in.
3. Press **Send**. It reaches the SOC queue as a New alert — nobody forwards an email.

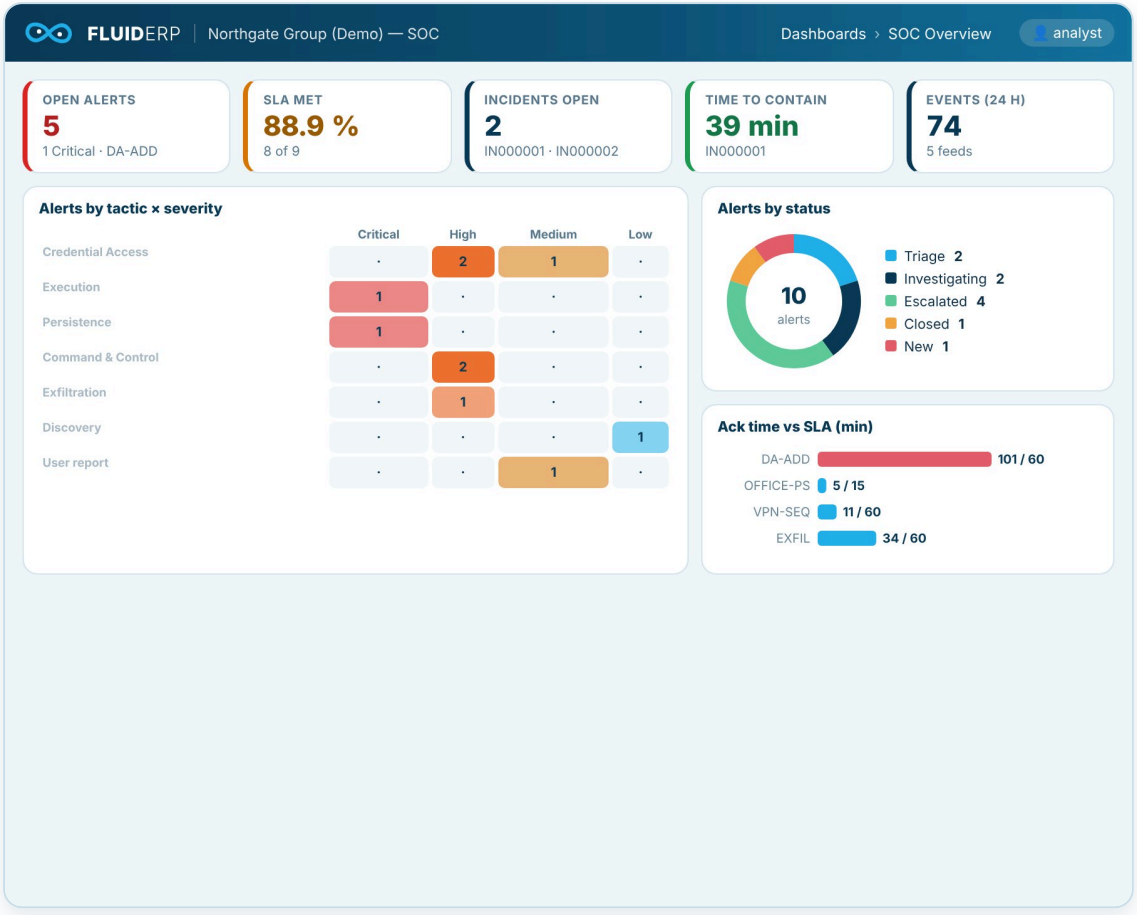
8 Critical dashboards

Three dashboards answer what is happening, where the rules cover, and which systems carry the risk.

SOC overview

Where **Dashboards › SOC Overview** Who **analyst, soc_lead**

Open alerts by severity, SLA, incidents and time to contain, alerts by tactic × severity and acknowledgement times.



SOC overview

What each tile is and where it comes from

Open alerts
Alerts not Escalated or Closed.
`secalert.status`

SLA met
Acknowledged within the severity's minutes.
`secalert.ack_in_sla`

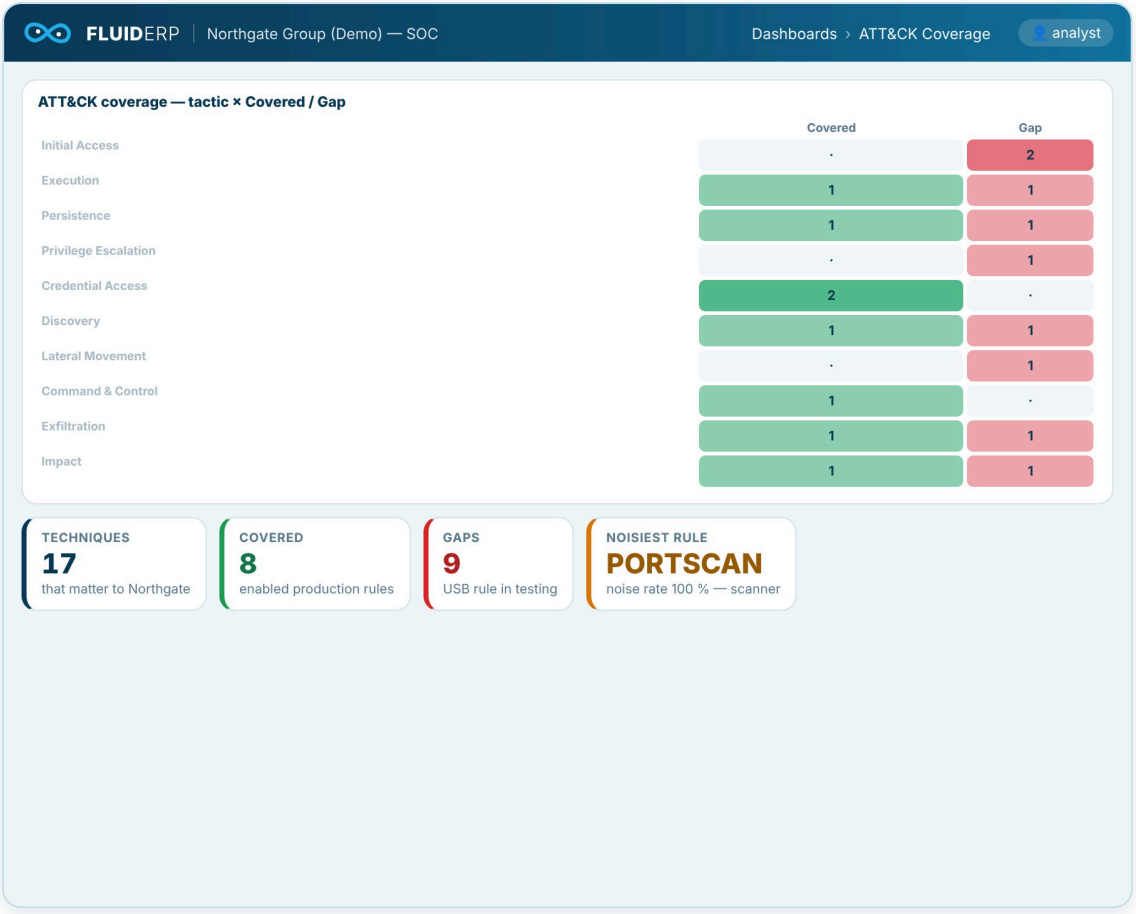
Time to contain
Contained at − opened.
`secincident.ttc`

Tactic × severity
Heatmap of alerts.
`secalert by tactic, severity`

ATT&CK coverage

Where **Dashboards › ATT&CK Coverage** | Who **soc_lead**

Which of the techniques that matter have an enabled production rule.



ATT&CK coverage

What each tile is and where it comes from

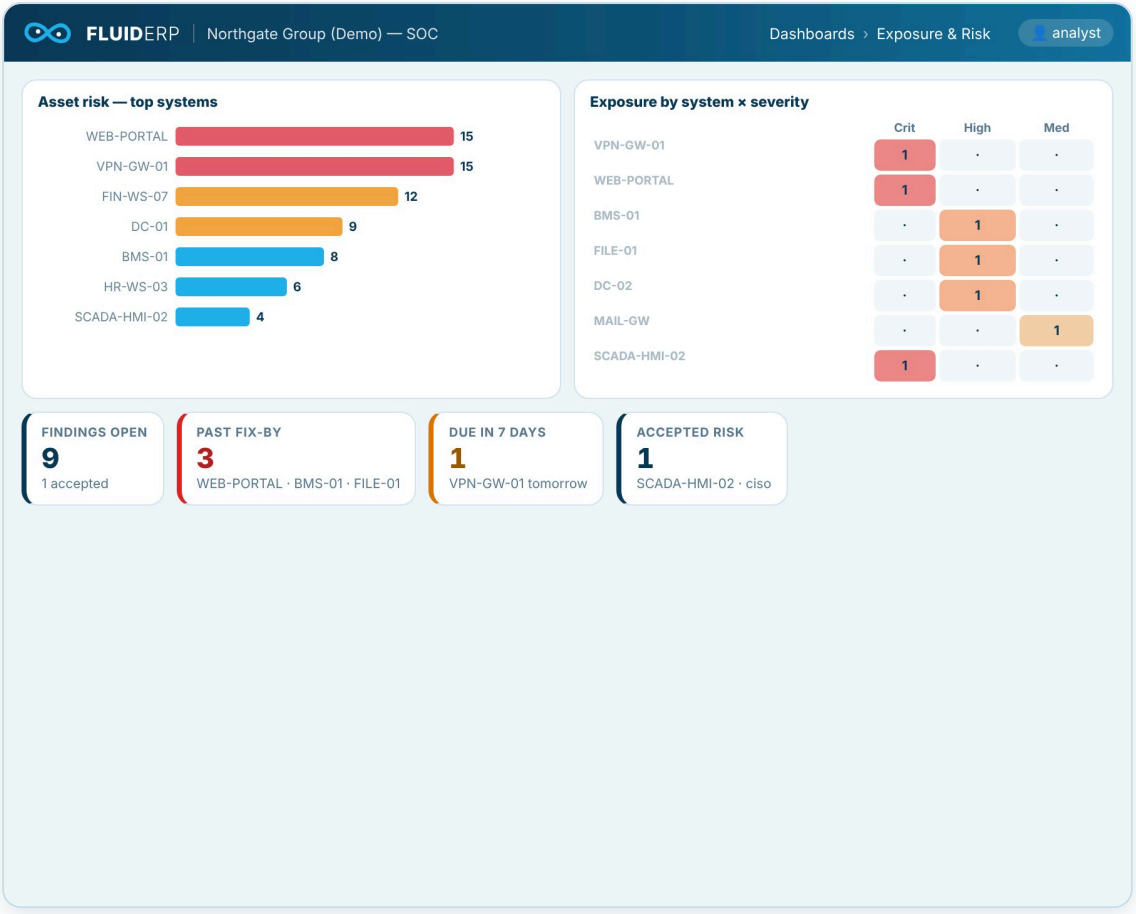
Covered / Gap
Per tactic.
`secattack · enabled secrule by technique`

Noisiest rule
Highest share of benign closures.
`secrule.noise_rate`

Exposure and risk

Where **Dashboards › Exposure & Risk** | Who **grc, ciso**

Asset risk and exposure by system × severity.



Exposure and risk

What each tile is and where it comes from

Asset risk
Criticality × (critical vulns × 3 + open alerts × 2 + open vulns).
`secasset.risk`

Past fix-by
Findings overdue.
`secvuln.overdue`

Accepted risk
Accepted by the CISO with a reason.
`secvuln.status`

9 Reports

Eleven SOC reports ship with the edition, alongside the finance and service-desk reports of the scaffold.

Running and printing a report

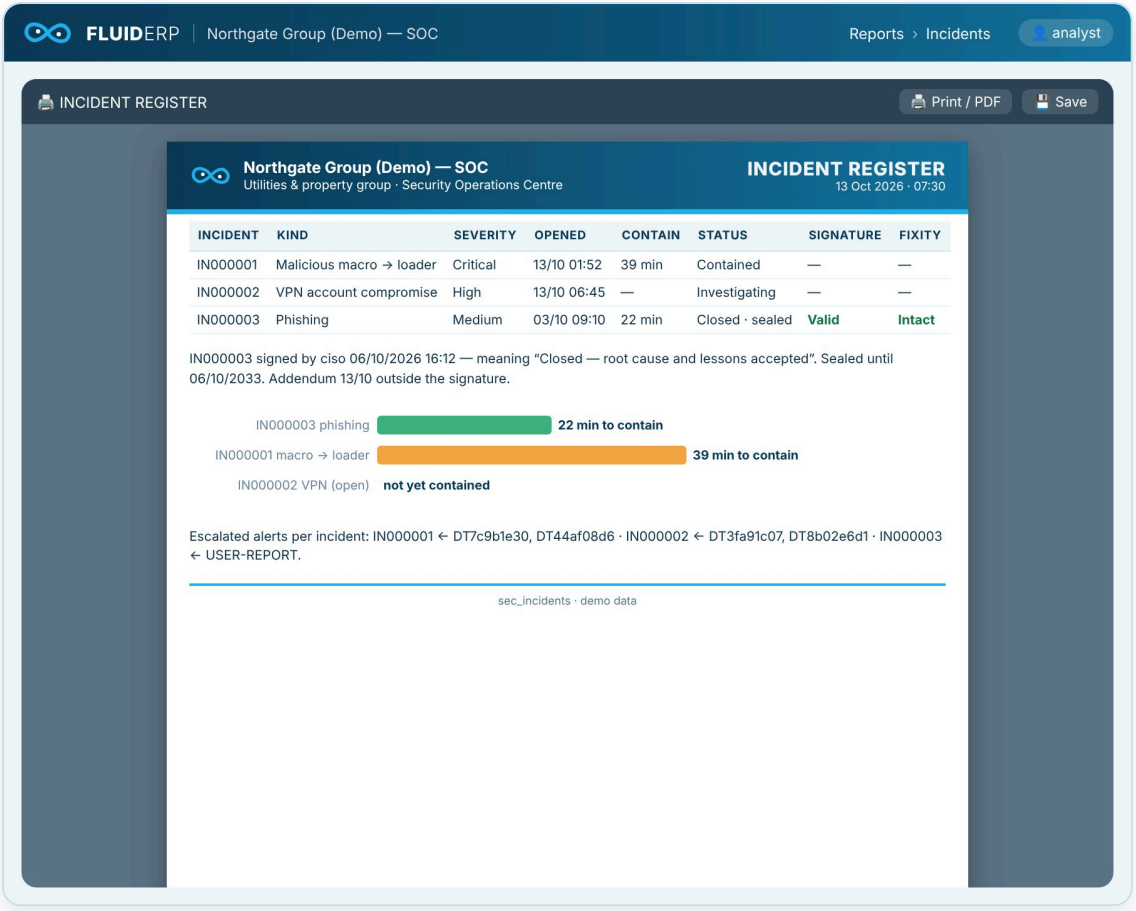
1. Open 🖨️ **Reports** on the launcher (or the report's own tile) and pick the report.
2. Fill the parameters. Required parameters are marked; dates accept `$today`-style defaults and lookups show "value — description".
3. Press ▶ **Run**. The report draws its shape — grid with subtotals, chart, pivot, calendar, Gantt, board or tree — from the live rows.
4. Press 🖨️ **Print** to open the printed copy (letterhead, document header, footer) in the document viewer, then 🖨️ **Print / PDF** or 💾 **Save**. Reports with a file layout also offer 💾 **Export File**.

Report catalogue

REPORT	SHOWS	TYPICAL READER
Alert Queue	Alerts by status, severity and SLA	analysts
Alerts by Tactic × Severity	Heatmap	soc_lead
ATT&CK Coverage	Covered / Gap by tactic	soc_lead
Rule Quality	Alerts, escalations and noise per rule	soc_lead
Incidents	Register with signatures and fixity	ciso, auditors
Threat Indicators and Hits	Indicators in force and their matches	analysts
Events by Host × Type	Volume heatmap	analysts
Vulnerabilities Past Their Date	Overdue findings by owner	grc, managers
Exposure by System × Severity	Open findings heatmap	grc
Control Posture	Maturity, evidence, tests	board, audit
Asset Risk	Systems ranked by risk	ciso

Incident register

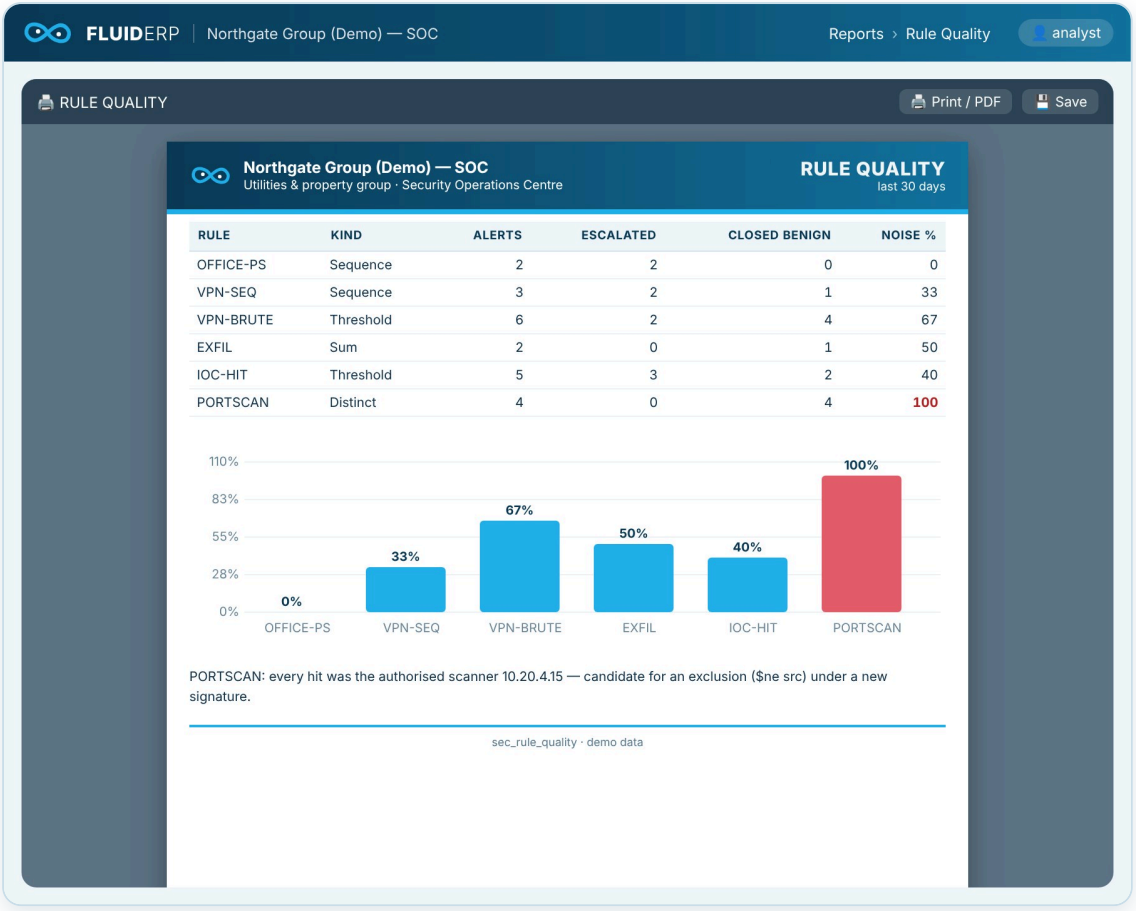
Time to contain and close are formulas; the signature status (Valid) and the fixity check (Intact) print beside each sealed record.



Incident register

Rule quality

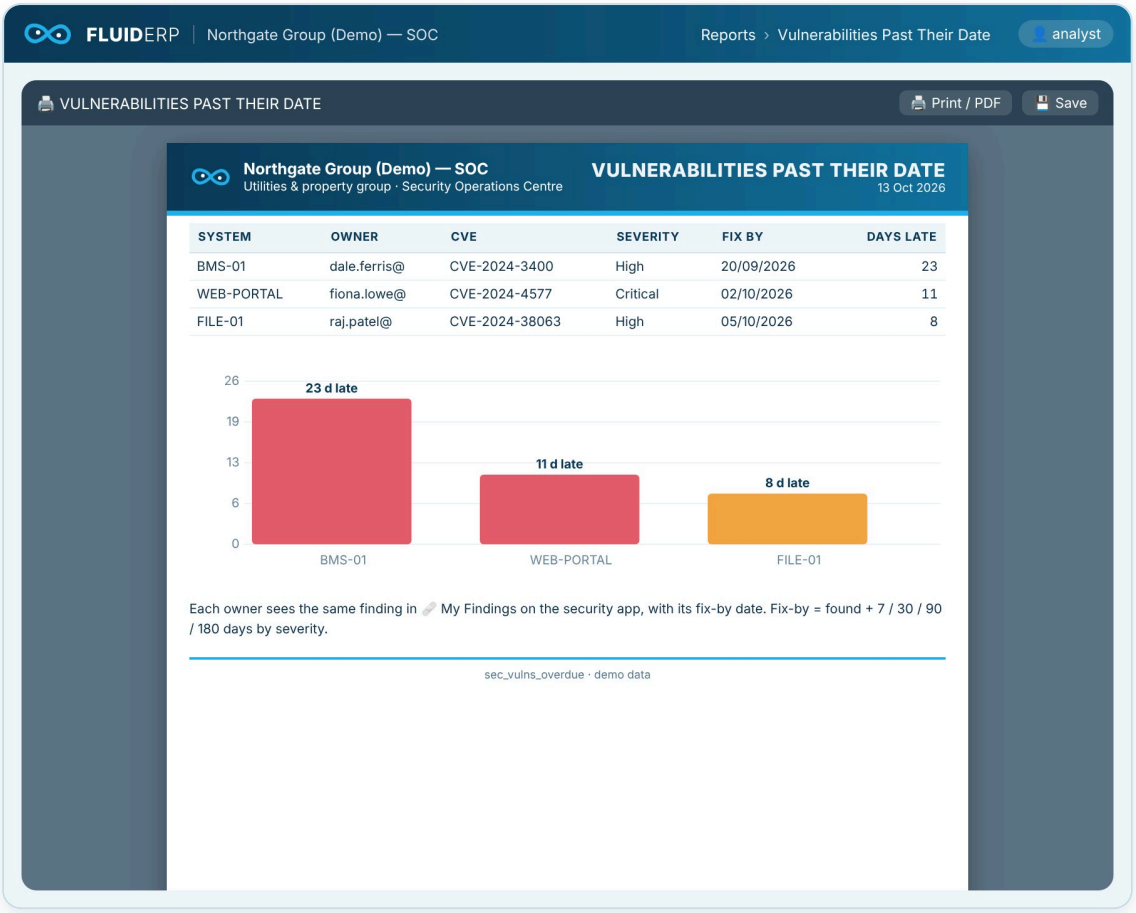
Alerts raised, escalated and closed benign per rule, with the noise rate. Tune from this; each tune is a new signature.



Rule quality

Vulnerabilities past their date

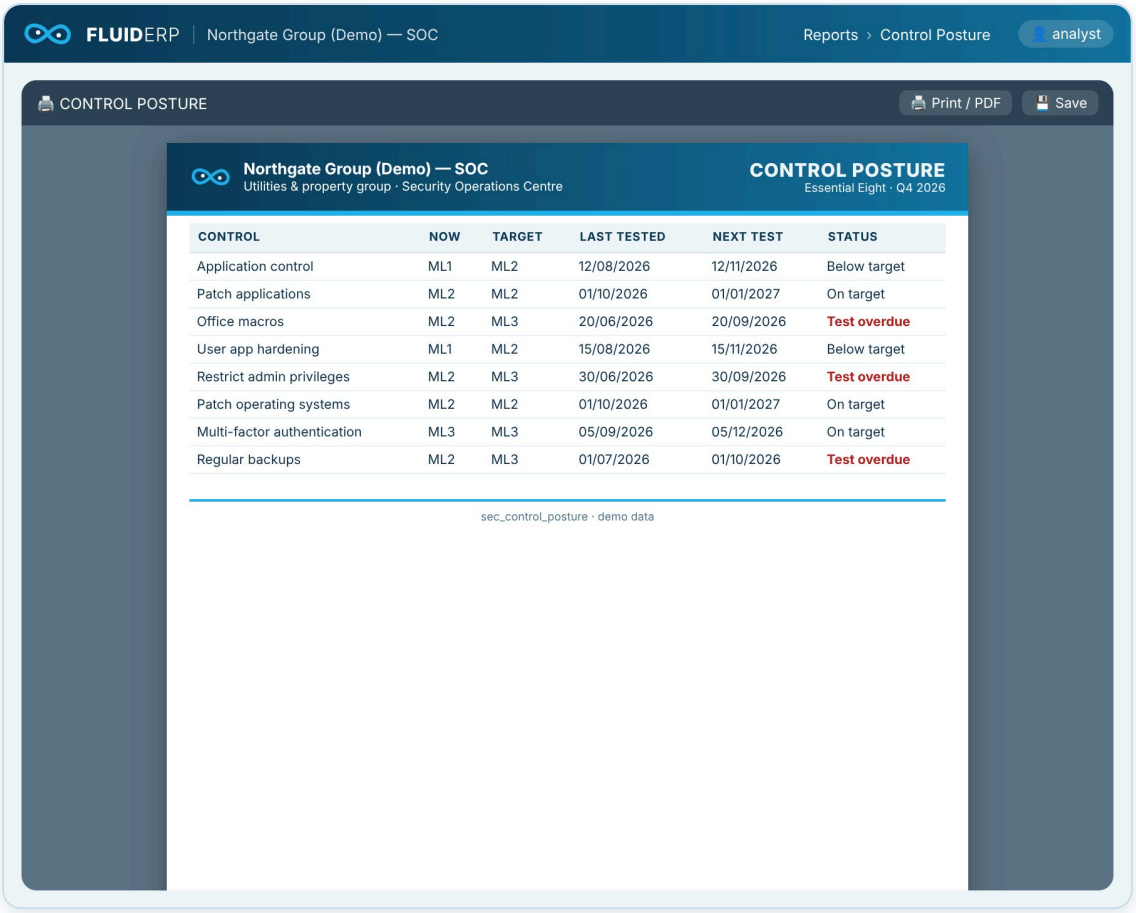
For the owners' managers: system, owner, finding, the date it was due and how late it is.



Vulnerabilities past their date

Control posture

Maturity against target, evidence and test dates for the audit committee.



Control posture

10 Matching

A matching screen sets two lists side by side — for example bank statement lines and cashbook entries — and pairs the rows that belong together. Matching is a definition (`tblmatchdef`): it names the two sides, the fields to compare, the tolerances and what a confirmed match writes.

How every matching screen works

1. Open the matching screen from its 🔀 tile on the launcher, or with 🔀 **Match** on a form that takes part in it.
2. 🔄 **Auto Match** runs when the screen opens. Rows that meet the rule are paired and arrive **pre-ticked** in green.
3. Read the **summary strip**: how many pairs, how many unmatched on each side, and any outstanding amount.
4. Untick any pair you do not accept. To pair rows by hand, tick one row on each side and press 🔗 **Match Selected**. ✖ on a pair puts both rows back.
5. Press ✔ **Confirm Ticked**. Both sides are stamped (so they never match again), a session record is written, and the reconciliation report opens in the document viewer.

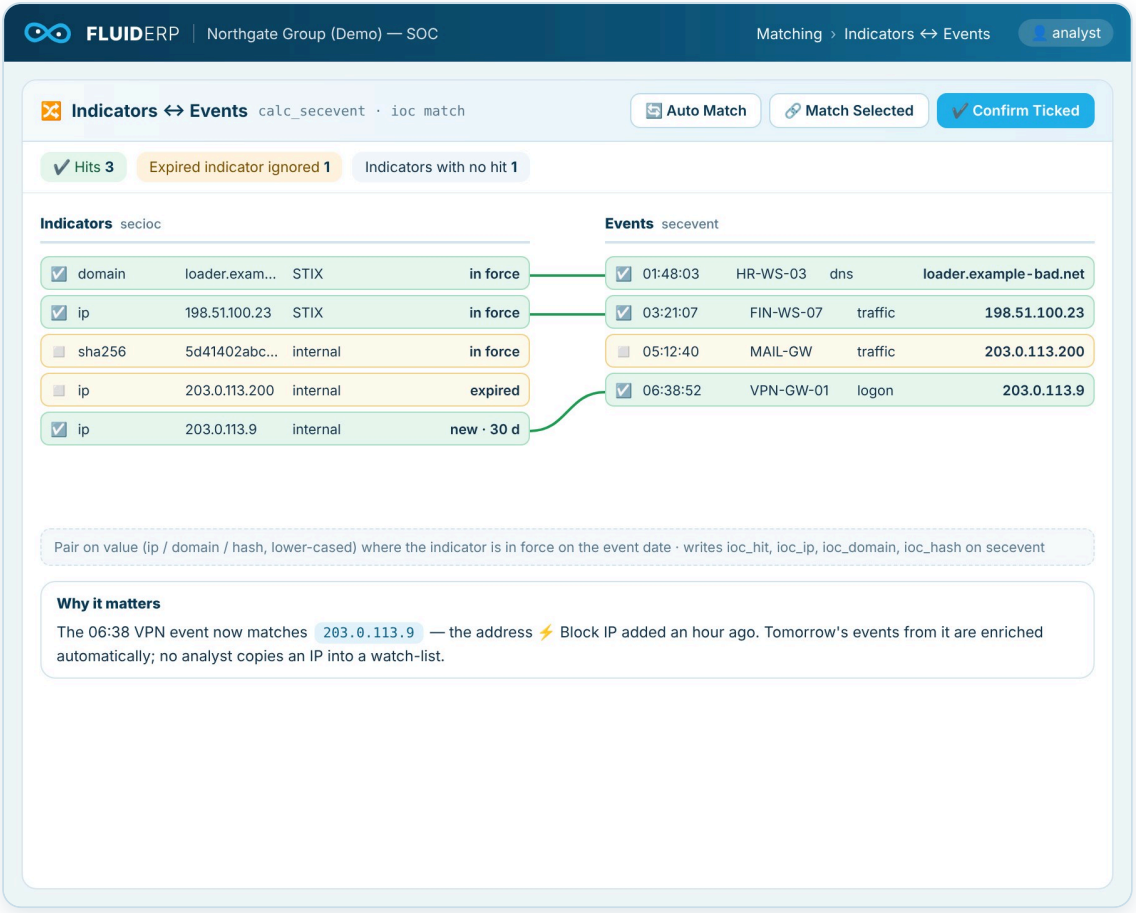
What ✔ Confirm Ticked writes	matching definition · confirm
✓ `sysmatch`	the session header — definition, who, when, number of groups (MAT number)
✓ `sysmatchlines`	one line per row in each confirmed group
✓ `each side`	the stamp field (for example matched_ind = M) and any counterpart fields the definition copies across

UNMATCHED IS INFORMATION

What is left after Auto Match is the exception list: an unexplained bank fee, a payment with no reference, a count that disagrees. Deal with those, not with the hundreds of rows that matched.

Indicators against events

Who **analyst** | Where **Matching › Indicators ↔ Events**



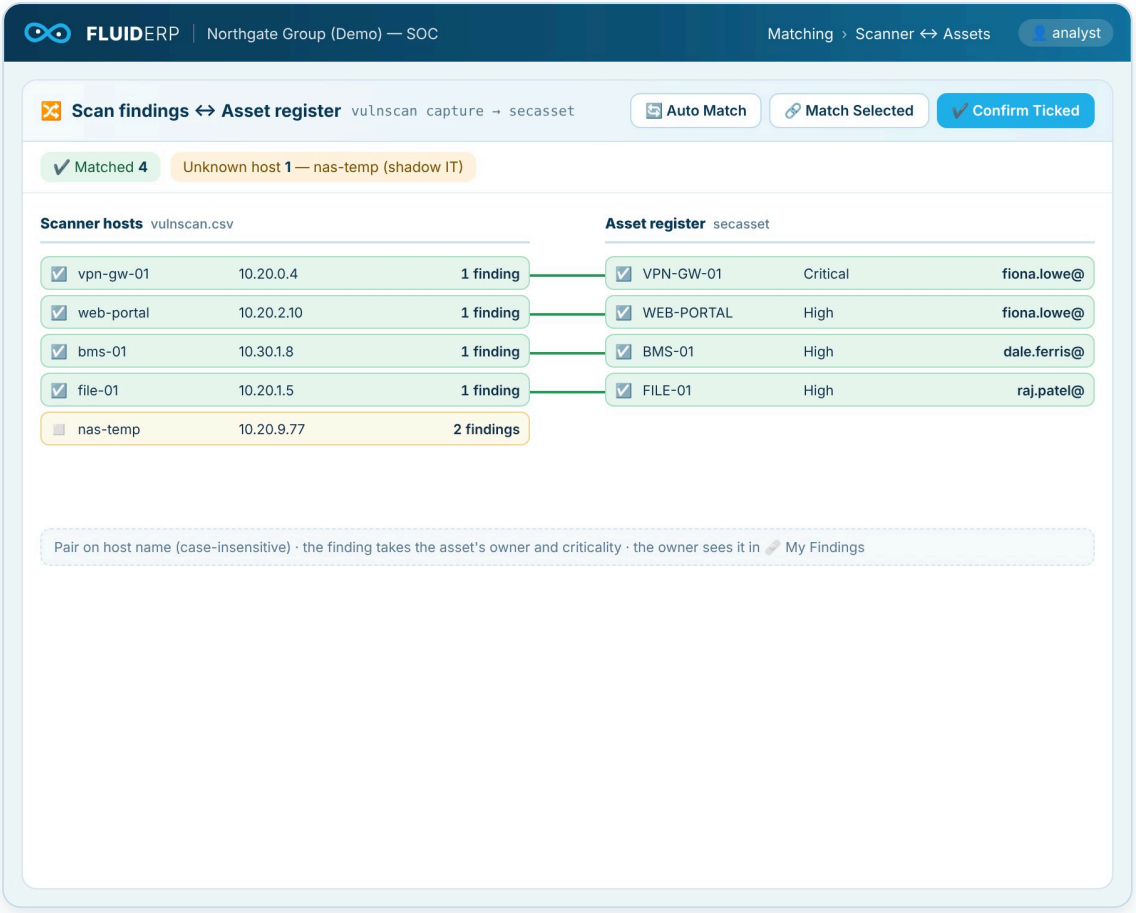
Three hits; the expired indicator is ignored; the newly blocked address already matches.

1. Open the screen to see the calculation's pairing: indicator value against event field, where the indicator is in force on the event date.
2. Unmatched indicators are candidates to retire; hits are what the IOC-HIT rule fires on.

> **NOTE**
> This pairing is done by calc_secevent; the screen presents it in the matching layout for review.

Scanner hosts against the asset register

Who **grc** | Where **Matching › Scanner ↔ Assets**



Four hosts match registered assets; nas-temp is unknown — shadow IT.

1. Findings pair with assets on host name (case-insensitive) and take the owner and criticality.
2. Register or decommission any host the register does not know.

11 Messages, refusals and what to do

FLUID refuses an action rather than let a figure go wrong. Every refusal names its rule. These are the ones you will meet on this platform.

MESSAGE (IN SHORT)	WHY	WHAT TO DO
Rule did not parse	Bad filter JSON or steps.	Fix the rule; others still run.
Signature superseded — reason and password required	A bound field of a signed rule changed.	The SOC lead re-signs.
A disposition is required to close	Alerts close with a reason.	Choose a disposition.
Disable Account refused — no user entity	The alert is about a host or address.	Use Isolate Host or Block IP.
An action must be Approved before Done	Response actions are approved instructions.	SOC lead approves.
Record sealed until …	Closed incidents are sealed for seven years.	Add an addendum.
Only the CISO closes an incident	Closure is the CISO's signature.	Ask the CISO.
Events are append-only	Evidence cannot be edited.	Capture a corrected event if needed.

12 Data reference and glossary

Tables you will meet

TABLE	HOLDS	WRITTEN BY
`secasset`	Assets, criticality, owner, risk	grc; calculations
`secevent`	Normalised events (append-only, full text)	capture definitions
`secrule`	Detection rules and their signatures	analysts; soc_lead
`secalert`	Alerts with SLA and disposition	the detection worker; security app
`secincident`	Incidents, signed and sealed	🚨 Escalate; responders; ciso
`secaction`	Approved response actions	⚡ buttons
`secioc`	Indicators (STIX and internal)	stixfeed; ⚡ Block IP; 🧠 Add to Intel
`secattack`	ATT&CK techniques and coverage	calculations
`secvuln`	Vulnerability findings	vulnscan capture
`seccontrol`	Controls and tests	grc
`tblcapture`	Capture definitions (formats and maps)	administrator

Glossary

TERM	MEANING
SOC	Security operations centre.
CEF	ArcSight Common Event Format.
STIX	Structured Threat Information Expression — indicator exchange format.
IOC	Indicator of compromise (address, domain, hash ...).
Episode	The set of events that satisfied a rule inside its window; one episode = one alert.
ATT&CK	MITRE's catalogue of adversary tactics and techniques.
Essential Eight	The Australian Signals Directorate's baseline mitigation strategies.
Fixity	A stored hash that proves a sealed record is unchanged (Intact).
Capture	Recording something once — by form, phone app, file or feed.
Posting button	An aqua button that writes rows to other tables from the record in front of you.
Posted · sealed	The record has been posted; the button is one-way and a second press writes nothing.
Calculated field (ƒ)	A read-only field the calculation engine keeps up to date.
Reference	The source record number carried on every row a button writes.

TERM	MEANING
Matching definition	The rule for pairing rows of two tables and what confirming writes.
Session (MAT…)	The record of one confirmed matching run.
Gateway	The server that checks every read and write against the rules for your group.

A Walk it once on the demo package

The Northgate package is installed at 07:30 on 13 October 2026 with the night's events, rules and alerts in place.

Analyst

1. Alerts › Queue: nine alerts; open DT3fa91c07 (VPN sequence) and read the five events.
2. Events: find the two ioc_hit rows (HR-WS-03 and FIN-WS-07).
3. Run detections as a dry run, then for real: created 0, existing 9.

Responder and SOC lead

1. IN000002 → ⚡ Disable Account and ⚡ Block IP; the lead approves; mark Done.
2. Try ⚡ Disable Account on an alert with no user — refused.
3. Rules: lower a Production rule's threshold — a new signature is demanded.

CISO

1. IN000003 (sealed): verify the signature (Valid) and fixity (Intact); try to edit — refused; add an addendum.
2. Vulnerabilities: accept SCADA-HMI-02's finding with a reason.

For the administrator

TASK	HOW
Deploy	`./deploy-secops.sh "<staff URI>" "<owner URI>"` .
Schedules	Detection worker every minute (`dryRun=false`); calc_secevent every 5 minutes (already `*/5 * * * *`).
Feeds	`POST /api/capture/{fwcef|edrjsonl|vpnsyslog|stixfeed|vulnscan}` ; samples in institutions/secops/samples.
Retention	Decide how long secevent rows stay online; events named by an incident are evidence.