



HANDS-ON WALKTHROUGH · USER ACTIVITY MONITORING (BETA)

Who is doing what on your Splunk?

One tenant watches every account of a deployment — people and service accounts — through the audit trail and the introspection, raises findings from a policy catalogue, and gives you the investigation workflow. From an empty tenant to classified accounts and findings.



You know what the data does. Do you know what the people do?

Data monitoring sees the feeds. A Splunk platform is also a few hundred saved searches, twenty accounts and the automation behind them — and their behaviour has no owner.

Schedules nobody reviews

Reports scheduled under a personal account, searches dispatched all time, a real-time schedule left running. Nobody notices until the owner leaves — or the search head does.

People and services look alike

A SOAR connector, a reporting script, an analyst: the same audit lines. Without a class, every policy is either noisy or blind.

Resource hogs are invisible

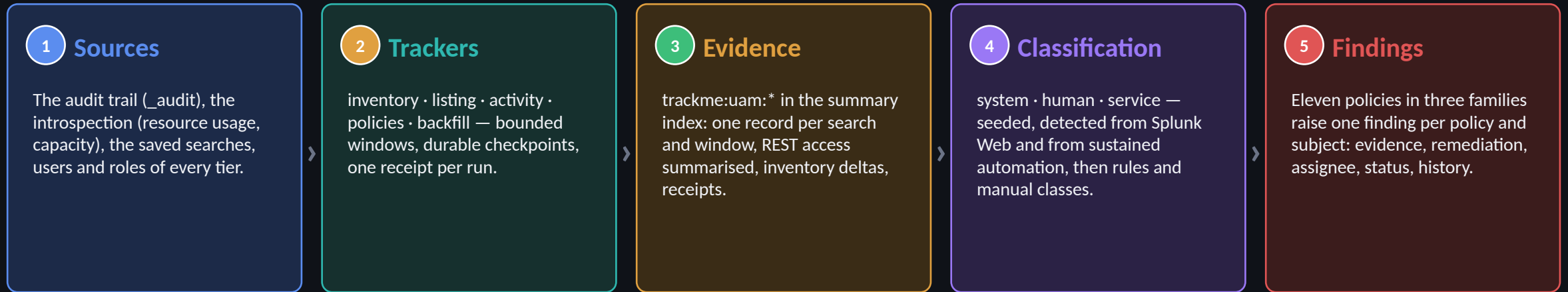
One account re-running a two-week search every five minutes takes half of the search CPU. The symptom is a slow search head; the cause is a search ID in the introspection.

What User Activity Monitoring does

UAM is a dedicated tenant type. It reads the audit trail and the introspection of a deployment, keeps an inventory of the scheduled searches, the users and the roles, records one compact evidence record per search and window, classifies every account as system, human or service, and evaluates a catalogue of eleven policies that raise findings — each with evidence, a suggested remediation, an assignee, a status and a history. It never edits or dispatches the searches it observes.

Beta in 2.4.18: a complete tenant type, captured here on a Splunk Cloud search head cluster — the platform you are about to see.

One tenant, one deployment: trackers, evidence, classes, findings



- 1 A dedicated tenant type**
Not an entity component: its own wizard card, its own tabs, its own trackers — one tenant per deployment.
- 2 One indexing layer, one or more tiers**
The local search head, plus any remote account as another tier of the same indexing layer.
- 3 Five trackers**
Inventory · listing · activity · policies · backfill. Bounded, checkpointed, replayed when a run dies.
- 4 Compact evidence, indexed**
trackme:uam:* — one record per search and window; the REST access of an account summarised per batch.
- 5 Classes, policies, findings**
Accounts are system, human or service; eleven policies raise one finding per policy and subject; you investigate.

Investigation on top: assign, investigate, resolve, dismiss — with notifications on the transitions and an AI Findings Advisor.

What we are going to do — one tenant, four parts

PART 1

Setup

- Pick the UAM card in the tenant chooser
- Basics, Configure UAM, Indexes & RBAC, Review
- A dedicated service-account owner
- The tenant card with its Beta badge

PART 1 · DAY ONE

Inventory, activity, first findings

- Five trackers scheduled at creation
- Collection health: receipts and backfill
- The tenant overview: KPIs and transitions
- The first findings are inventory facts

PART 2

Read & classify

- The Findings tab and the Investigate view
- The Users tab: human · service · system
- One account deep-dive, the Activity tab
- How the classification is decided

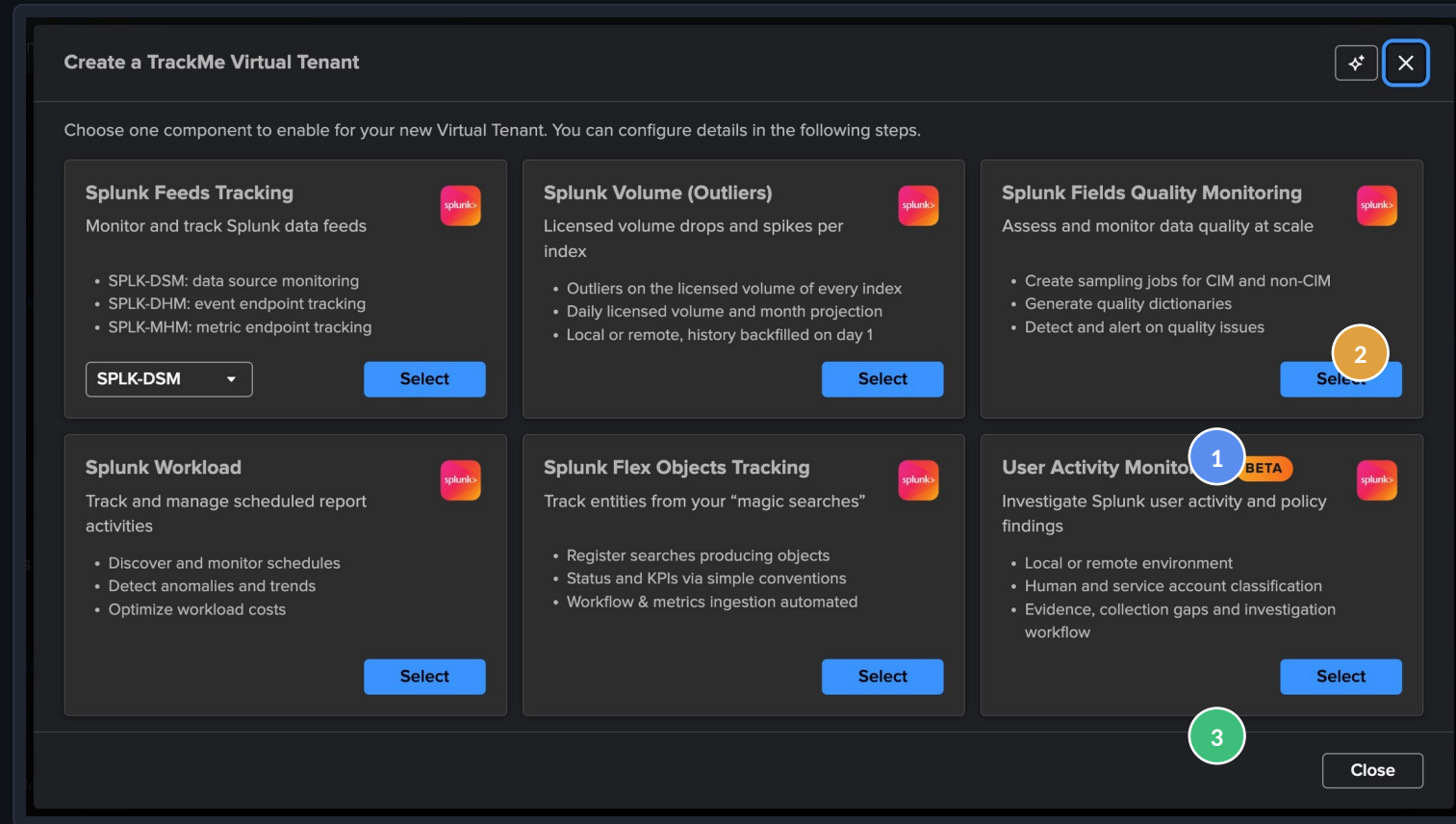
PART 3 · 4

Tune & operate

- The policy catalogue and the insight
- Classification rules, collection limits
- Notifications: digest or threads
- Collection health, recovery, the Guardian

One Virtual Tenant, one local search head tier, a 30-day backfill. Every screen is the product as shipped; nothing was pre-configured.

A dedicated tenant — pick User Activity Monitoring in the chooser



1 The User Activity Monitoring card

The sixth card of the chooser, next to the five entity components. One tenant monitors one deployment.

2 Beta badge

Introduced in 2.4.18 as a Beta: a complete tenant type whose policies and evidence may still move between beta releases.

3 Select

Opens the UAM flavour of the tenant wizard: Basics, Configure UAM, Indexes & RBAC, Review.

Not an entity component

A UAM tenant carries no entities: no DSM, DHM or FLX component can be added to it, and it does not appear in the entity views. It has its own tabs — Findings, Users, Activity, Scheduled searches, Policies, Collection health, Settings.

Basics — an identifier, an alias, and the Beta notice

Create a new TrackMe Virtual Tenant: User Activity Monitoring **BETA**

1 Basics 2 Configure UAM 3 Indexes & RBAC 4 Review

Provide an identifier for your Virtual Tenant. This cannot be changed later.

Tenant identifier

uam-cloud

Lowercase letters, digits and hyphens; up to 20 characters.

Tenant alias (optional)

User activity

Tenant description (optional)

User Activity Monitoring of this Splunk Cloud stack

BETA **User Activity Monitoring is in beta**

This is an early release. Its screens, policies and settings may still change as we learn from the first customers using it, and your feedback directly shapes what comes next. Your tenant, its configuration and its findings are kept across those changes.

It investigates Splunk user activity and raises policy findings. A license beyond the Foundation edition is required (Enterprise, Unlimited, trial or developer); audit evidence is restricted to tenant administrators.

< Back Next >

1 Tenant id — uam-cloud

Immutable identifier, checked for availability as you type; it names the KV collections and the trackers.

2 Alias and description

What people see on the Virtual Tenants card: "User activity monitoring" here.

3 The Beta notice

States what the beta does not do yet, and that the schema may move between beta releases.

How many tenants

One deployment, one UAM tenant. A second tenant would monitor a second indexing layer — another Splunk stack — not a second team of the same one; the RBAC of the findings lives inside the tenant.

Configure UAM — the tiers, the route account, the backfill

Create a new TrackMe Virtual Tenant: User Activity Monitoring BETA

✓ Basics

2 Configure UAM

3 Indexes & RBAC

4 Review

Select the search head tiers whose user activity you want to investigate.

☒ Local search head

1

The search head TrackMe runs on

Configure a TrackMe remote account to add another search head tier.

Remote accounts are tested automatically when selected; creation waits until every selected remote account is reachable.

ROUTE ACCOUNT

2

Local search head

The tier the Indexing-layer searches (activity, resource usage, capacity) go through; derived from the selection: the local search head when it is selected, else the first remote tier.

3

ONE INDEXING LAYER PER TENANT

A UAM tenant monitors one indexing layer through one or more search head tiers. Each tier is one account: the local search head TrackMe runs on, or a configured TrackMe remote account. On Splunk Cloud, for example, TrackMe on the ad-hoc search head is the local tier and a remote account adds the Enterprise Security search head.

Saved searches, users and roles are inventoried once per tier. Audit, introspection and internal data of every tier land in the same indexing layer, so activity, resource usage and capacity are collected once per tenant through the route account.

Every tier selected for a tenant must share the same indexing layer — that is the administrator's responsibility; a tier on another indexing layer belongs to another tenant.

HISTORICAL BACKFILL

Range of history to collect

30 days

Preset: 30 days. Changeable later in Settings → Collection limits.

Collects the audit and resource evidence of the recent past through the route account, bucket by bucket, hourly, until the range is covered by a report then retires itself. Speed depends on the deployment. Introspection retention is typically 14 days, so resource history may end earlier than audit history.

4

< Back

Next >

- 1 **Search head tiers — local**
The local search head is the first tier. Any remote account adds another tier of the same indexing layer (an ES cluster, for instance).
- 2 **Connectivity test**
Every remote tier passes its connectivity test before the creation is allowed.
- 3 **Route account**
The account used for the indexing-layer searches — audit, introspection, internal data. Derived: local here.
- 4 **Historical backfill — 30 days**
The audit and resource history collected before the live start. None, or up to 90 days.

Why backfill

The backfill is what makes the classification usable on day one: people are detected from the past weeks of Splunk Web activity, and service accounts earn their seven days of automation-only activity from history instead of waiting a week.

Indexes & RBAC — give the tenant a service-account owner

Create a new TrackMe Virtual Tenant: User Activity Monitoring BETA

Progress: Basics (1) ✓, Configure UAM (2) ✓, **Indexes & RBAC (3)**, Review (4)

Choose indexes for TrackMe objects and assign ownership and roles.

Summary index:

Audit index:

Metric index:

Notable index:

Splunk owner for created objects:

Admin roles:

Power roles:

Readonly roles:

< Back Next >

1 Indexes

Summary, audit, metric and notable indexes — the defaults are fine. The trackme:uam:* evidence lands in the summary index.

2 Owner — a dedicated service account

The recommendation specific to UAM: an account with administrative rights (svc-trackme). The trackers and the scheduled AI review run as the owner.

3 Roles

Admin, power and user roles of the tenant: who administers, who investigates, who reads the findings.

What the owner needs

The owner must be able to read `_audit` and `_introspection` and to list the saved searches, users and roles of every tier. An owner without those rights leaves the inventory and the activity empty — and the Configuration Guardian tells you.

Review — the summary, then everything is created in seconds

Create a new TrackMe Virtual Tenant: User Activity Monitoring BETA

Progress: Basics (✓) → Configure UAM (✓) → Indexes & RBAC (✓) → Review (4)

1 Please review the configuration before confirmation.

Update comment (optional)
Add a note for this creation

SUMMARY

User activity

Tenant identifier: uam-cloud

User Activity Monitoring of this Splunk Cloud stack

Search head tiers: **Local search head · route**

Route account: **Local search head** Owner: **svc-trackme**

Administrator roles: trackme

Historical backfill: **30 days** — collected hourly through the route account, bucket by bucket, from the live activity start backwards; the report retires itself once the range is covered.

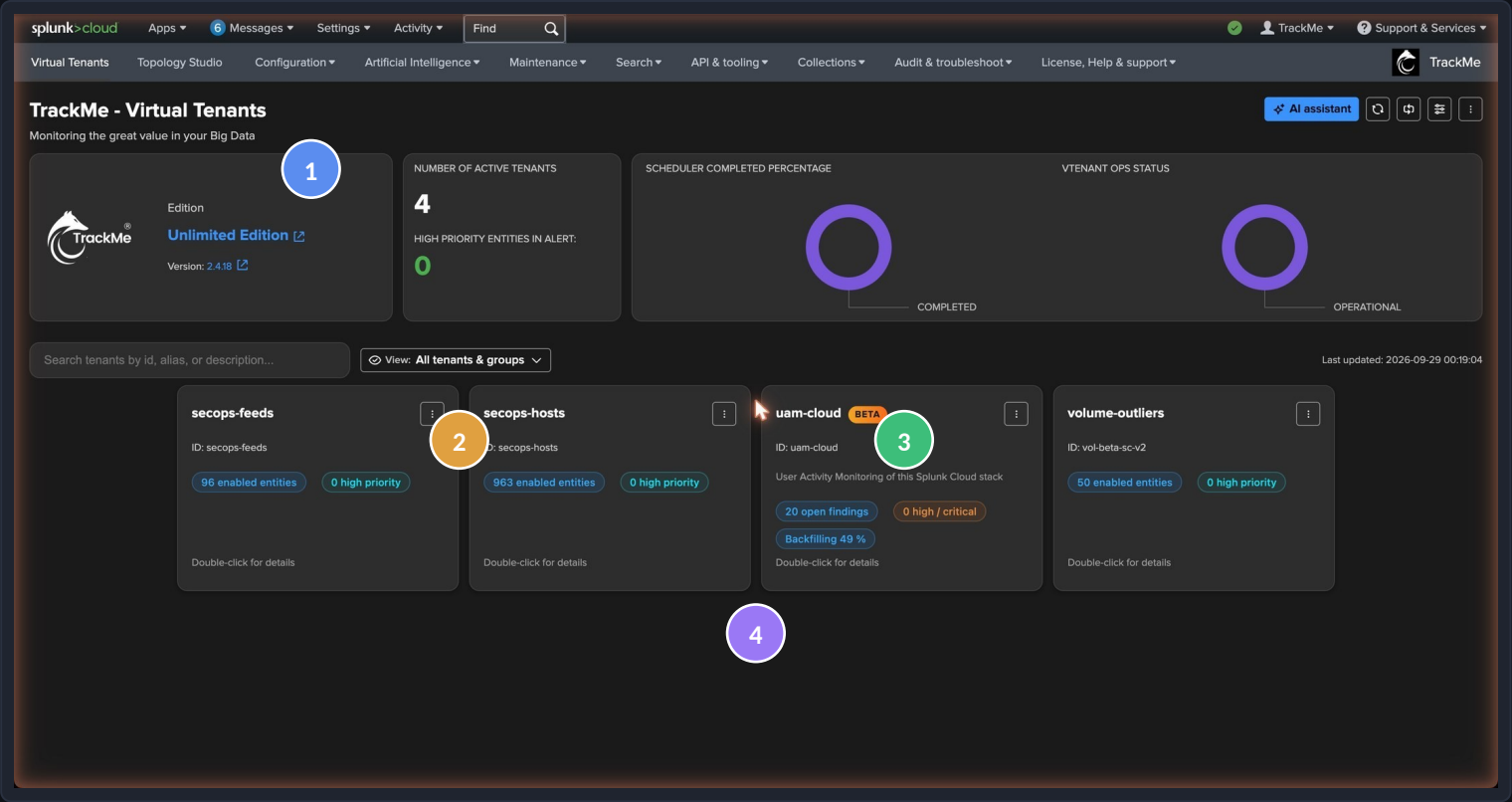
Trackers scheduled at creation: one inventory report per search head tier every 15 minutes (1 tier) with its full listing as a report of its own (its first listing within the hour, then daily in the off-peak window), activity every 5 minutes from a durable ingestion checkpoint through the route account, and policies every 15 minutes. In the workspace, classify accounts and review finding policies.

Evidence is indexed as trackme:uam:* events in the tenant summary and notable indexes; you can point each family at another index in Settings.

< Back Done >

- 1 Identity and tiers**
uam-cloud, one local tier, the route account, the 30-day backfill.
- 2 What gets created**
Three KV collections (settings, inventory, findings), the lookups, five trackers and the health tracker.
- 3 Scheduled immediately**
The inventory, listing, activity and policies trackers start at their next slot; the backfill plans itself at its first hourly slot.
- 4 Create**
Or automate it: one REST call to `vtenants/admin/add_tenant` with `tenant_uam_enabled` and a `uam_config` object.

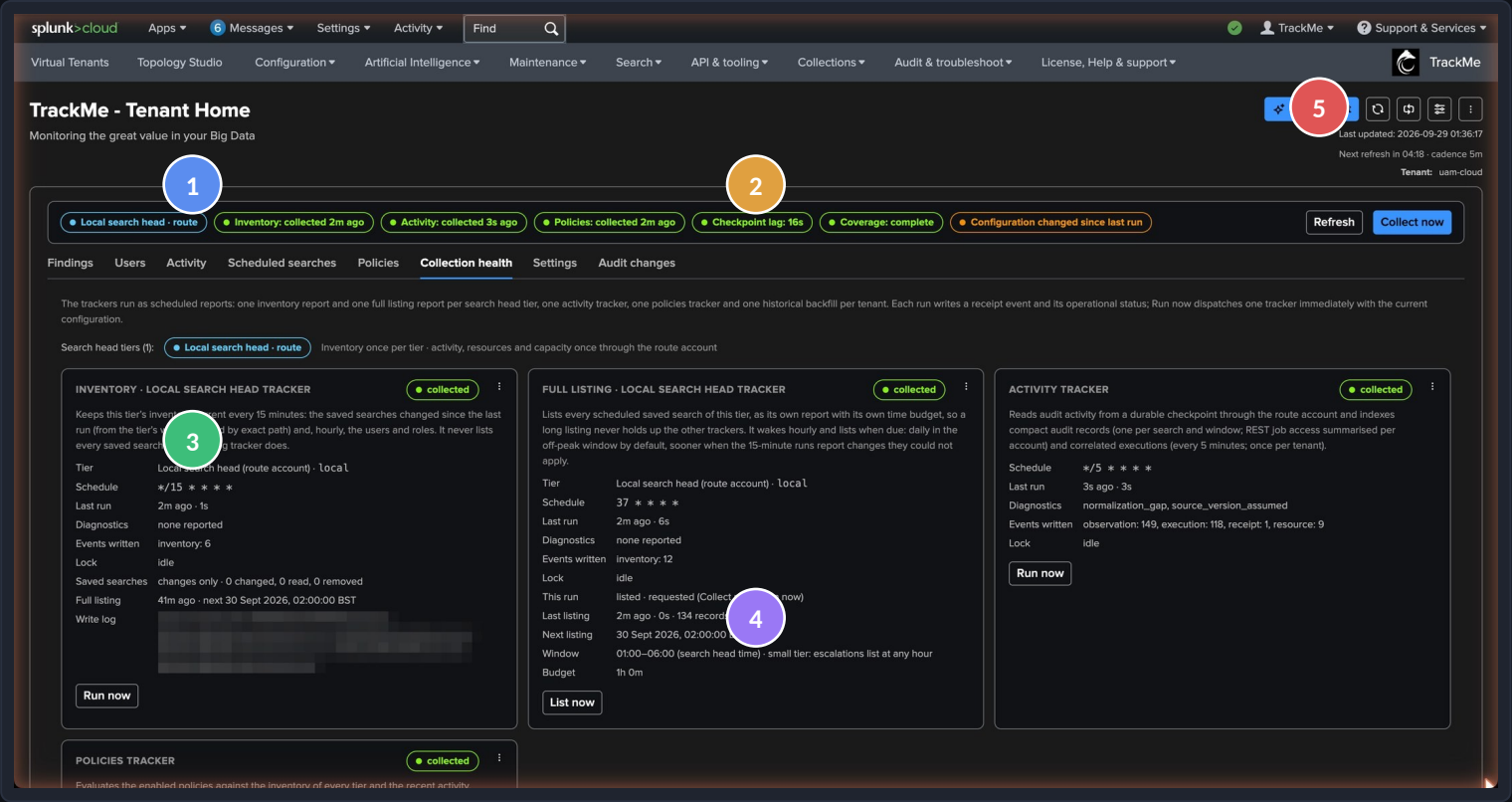
The tenant card — a Beta badge, findings and the backfill at a glance



- 1 **The UAM card**
Alias, identifier, the Beta badge and the tier list — no entity KPIs, no donut.
- 2 **Open findings pill**
Open findings by severity — empty right after creation, the inventory facts appear within the quarter of an hour.
- 3 **Backfill pill**
Planned, running with its progress, or complete: the history being walked backwards, newest first.
- 4 **Collection status**
Green while the trackers run on time; the Guardian check uam_collection_health turns it amber when something stalls.

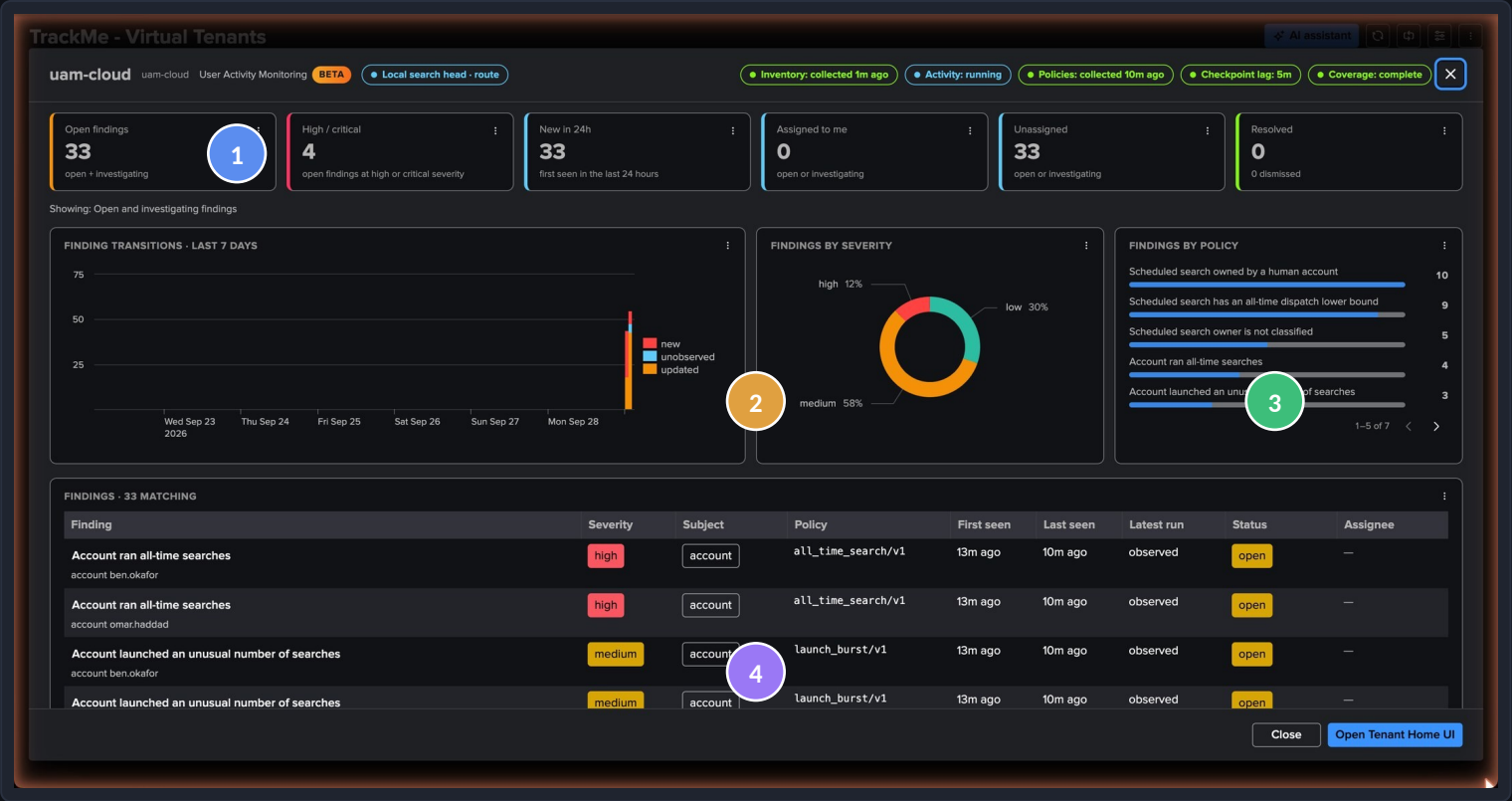
Virtual Tenants, a few seconds after creation.

Collection health — five trackers, their receipts, the backfill walking back



- 1 The trackers**
Inventory · listing · activity · policies · backfill, each with its last run explained and its seven-day history.
- 2 Receipts**
One trackme:uam:receipt per run: windows read, records written, gaps, duration.
- 3 Activity checkpoint and lag**
The audit trail is read five minutes at a time from a durable checkpoint — never all time.
- 4 Backfill progress**
Thirty days walked backwards in fifteen-minute buckets, a little under an hour per slot; its edges show where the history ends.
- 5 Collect now**
Dispatches every tracker on demand. A scheduled run never disables the page.

The tenant overview — first KPIs, first transitions, first policies



- 1 KPI tiles**
Accounts seen, scheduled searches inventoried, open findings by severity, evidence records of the day.
- 2 Transitions over time**
New, reopened, escalated, resolved, dismissed — the life of the backlog as a timeline.
- 3 By policy**
Which policies raised what: on day one, `all_time_schedule` and `realtime_schedule` — inventory facts.
- 4 No personal_schedule yet**
That policy needs the owners classified as human first; the backfill and the next part bring it.

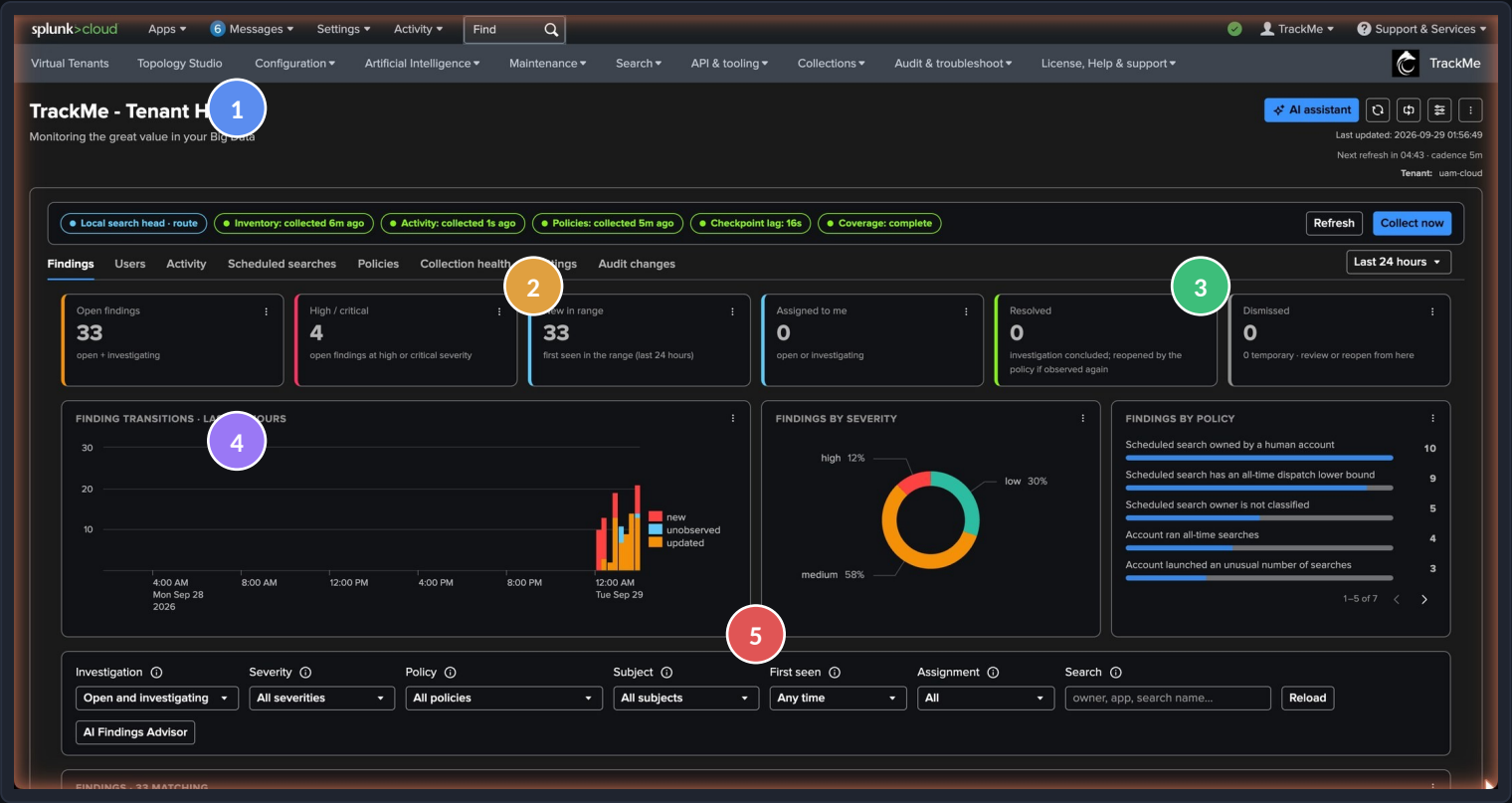
01

PART 2 · READ & CLASSIFY

Findings, accounts, evidence

The Findings tab and the Investigate view; the Users tab where the classes settle; one account deep-dive; the Activity tab with the raw evidence. Most of the classification happens without us.

The Findings tab — the backlog, one row per policy and subject



- 1 Tiles
Open, high / critical, new in the range, assigned to me, resolved, dismissed.
- 2 Transitions chart
The backlog over time by event; one bar per policies run that changed something.
- 3 By severity, by policy
Where the weight is: severity on the left, policy on the right.
- 4 Filters
Status, severity, policy, subject type, assignee, "still observed by the latest run".
- 5 The table
Severity, subject (a schedule or an account), policy, first and last seen, observed, status, assignee. Bulk edit on a selection.

Investigate a finding — subject, evidence, suggested action, decisions

Investigate finding

high · account · open

5

Account ran all-time searches

f41243a91a6fadcb4a8e5fb05166f4f8211104dd884cc457db68f8b41c9aca55

Copy id

Open record in search

Open events in search

AI Findings Advisor

Notified 0 seconds ago (1 notification)

2

SUBJECT

ACCOUNT

alice.moreau · human · detected

ENVIRONMENT

local

Open account

3

SEVERITY

high (potential)

LAST

29 Sept 2026, 01:48:29 BST (not observed 1 in latest run)

POLICY

Account ran all-time searches

EPISODES

FIRST SEEN

29 Sept 2026, 01:48:29 BST

4

Suggested action: Review the searches in the Activity tab and agree on bounded time ranges or an account exception for approved investigations.

6

EVIDENCE

Raw JSON

window.start · 29 Sept 2026, 01:33:29 BST

window.end · 29 Sept 2026, 01:48:29 BST

count · 1

max_all_time_searches · 0

account_class · human

classification.value · human

Close

Save investigation

- 1

Severity and status chips
high · open · account — alice.moreau ran an all-time search.
- 2

The subject
Here an account, its class and where it comes from (human · detected), its environment; for a schedule: the saved search, its app, its owner.
- 3

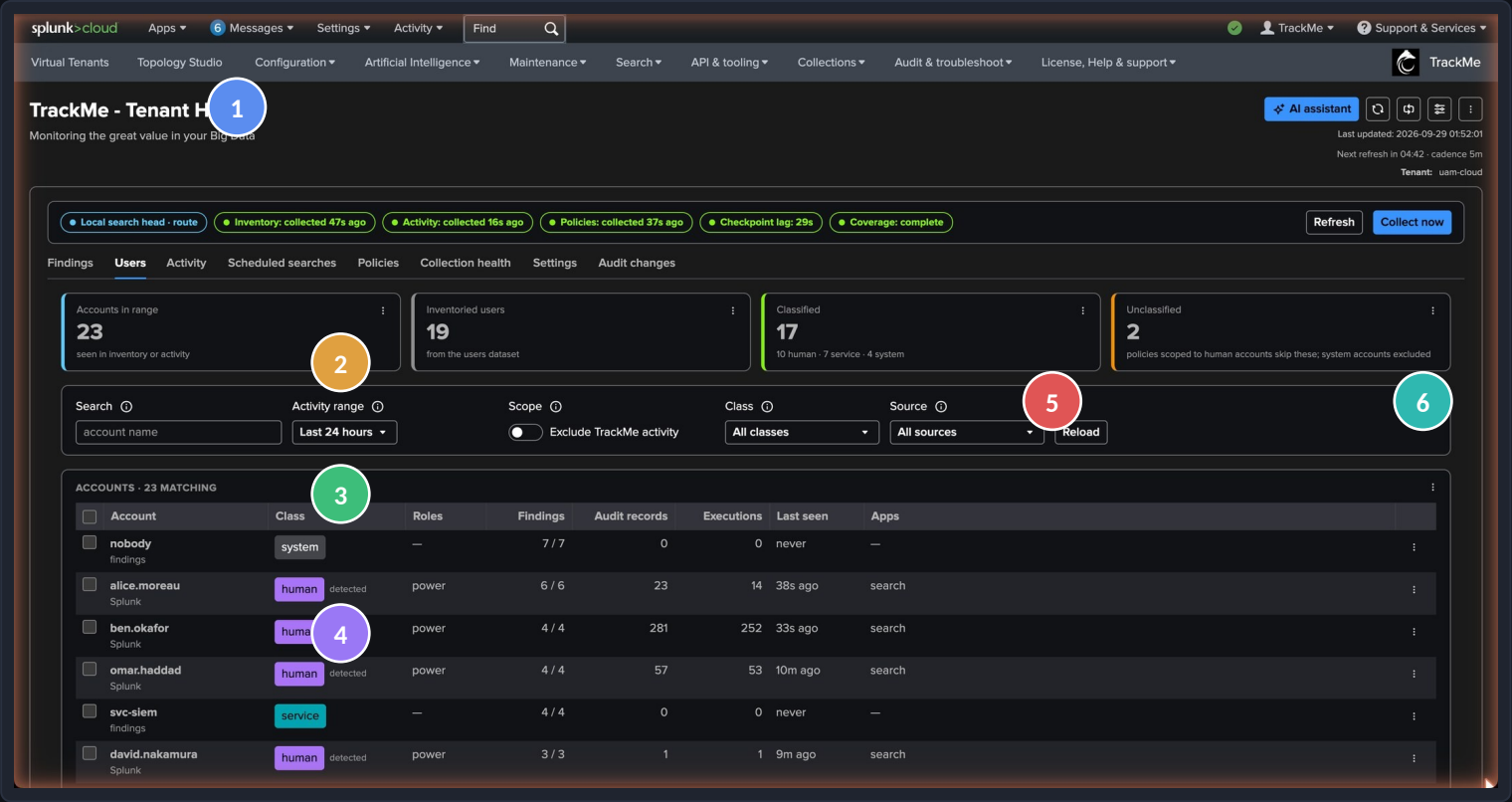
The evidence
The policy window, the all-time searches observed in it and the threshold in force (max_all_time_searches 0); for a schedule, the dispatch bounds read.
- 4

Suggested action
Advisory only: UAM never edits nor dispatches the observed searches.
- 5

Open events · record · owner account
Every indexed transition, the current KV record, and the deep-dive of the owner.
- 6

Assign, investigate, resolve, dismiss
Each decision with who, when and a note; a resolved finding reopens if the condition returns.
- docs.trackme-solutions.com

The Users tab — every account seen, with its class and its weight



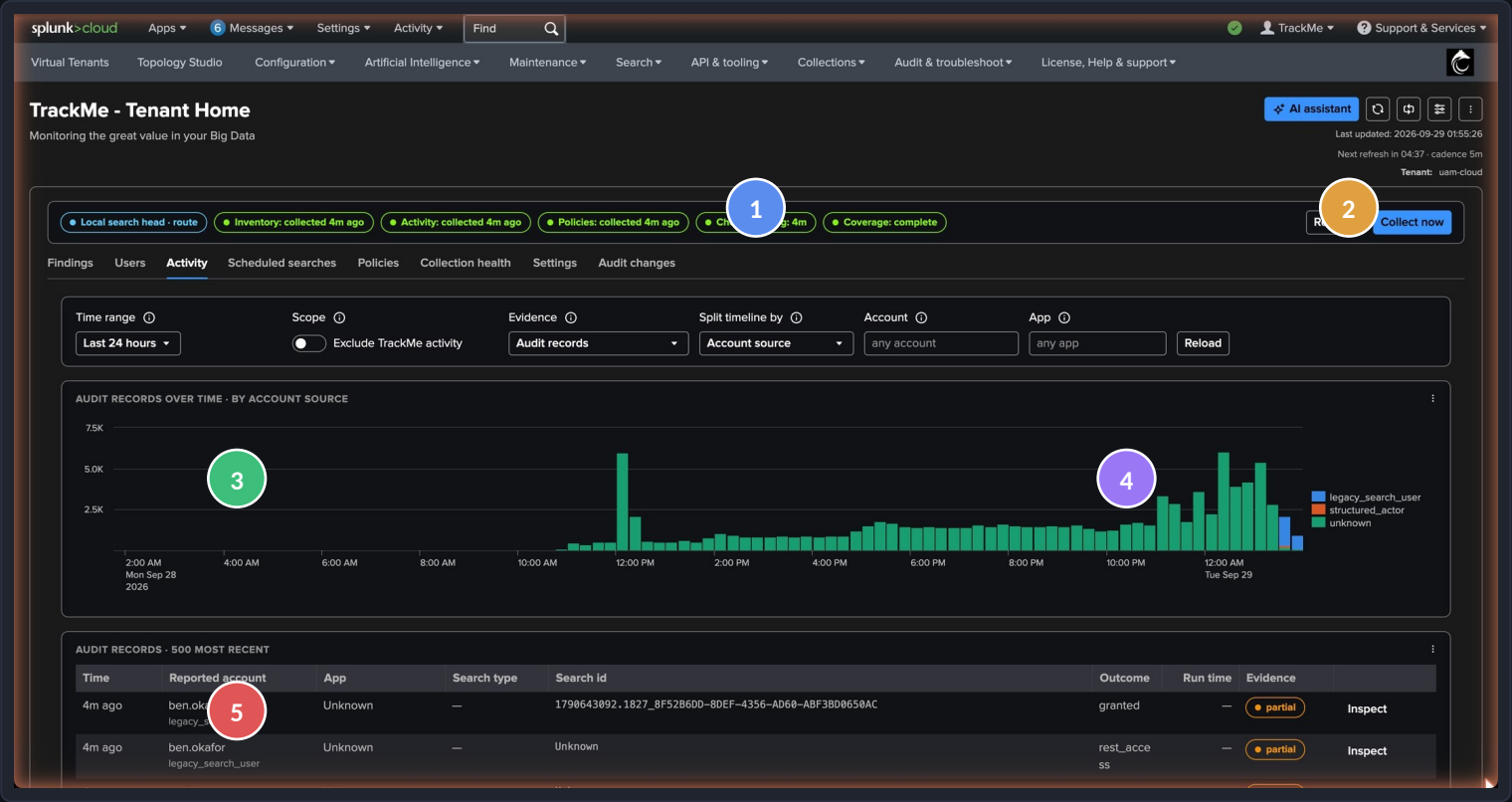
- 1 **Tiles**
Accounts, inventoried users, classified and unclassified.
- 2 **human • detected**
alice.moreau, ben.okafor, tom.hendricks: a Splunk Web search or a browser login. One interactive signal is enough.
- 3 **service • detected**
svc-siem-collector, svc-soar-actions: seven distinct days of automation only, or the svc-* rule on day one.
- 4 **system**
nobody, splunk-system-user, admin by default; svc-trackme and the administrators by our decision.
- 5 **Findings and evidence counts**
Findings, audit records and executions, last activity, apps — per account.
- 6 **Class chip actions**
The row menu classifies an account in place: svc-bi-exporter → service, before it earns its seven days.

One account — its activity, its resources, its findings



- 1 **Identity**
svc-bi-exporter — service by the svc-* rule; its roles, its apps, its tier.
- 2 **Activity**
Executions over time by search type: API dispatches only, no Splunk Web — the profile of an export job.
- 3 **Resources**
CPU timeline by search type, indexer share, peak memory, and the twenty most expensive searches.
- 4 **Findings**
Every finding where this account is the subject or the owner; the resource policies judge it against the capacity-derived allowance.
- 5 **Open in Search**
The rollup SPL of each panel, for the analyst who wants the raw view.

The Activity tab — the audit records, one per search and window



- 1 Records over time**
The audit records by account source: the UI, the REST API, the scheduler.
- 2 By account**
Who generates the records: the connectors dominate by count, the analysts by variety.
- 3 The records table**
Account, app, search id, type, status, run time, events and results, the time range searched.
- 4 Evidence pills**
One trackme:uam:observation per search and window; the REST access of an account summarised per batch.
- 5 Search lifecycle**
Dispatch, run and completion correlated on the search id, with the resource record of the introspection.

How an account gets its class — and in which order

human • detected

A search from Splunk Web (the UI: * provenance) or a browser login. One interactive signal is enough, and the class is sticky.

service • detected

Seven distinct UTC days of automation-only activity — rest:jobs, the scheduler — and nothing a person does. One Splunk Web search withdraws it.

rules and manual classes

Ordered exact or wildcard rules (svc-* → service) previewed on the known accounts; a manual class from the row menu overrides everything.

system

nobody, splunk-system-user, admin by default. Add the tenant owner and the administrators: the platform's own workload never appears in a finding.

Precedence

manual class → system defaults → first matching rule → human detection → service detection

Two limits of the beta

The automation scan reads the legacy key=value _audit format — a JSON-only _audit gives no automatic service detection — and the seven days must lie inside the audit retention. Both are stated in the Beta notice; a rule or a manual class covers the gap.

What changes next

Once the owners are classified, the next policies run raises the personal_schedule findings — at severity low, by design: hygiene, not incidents.

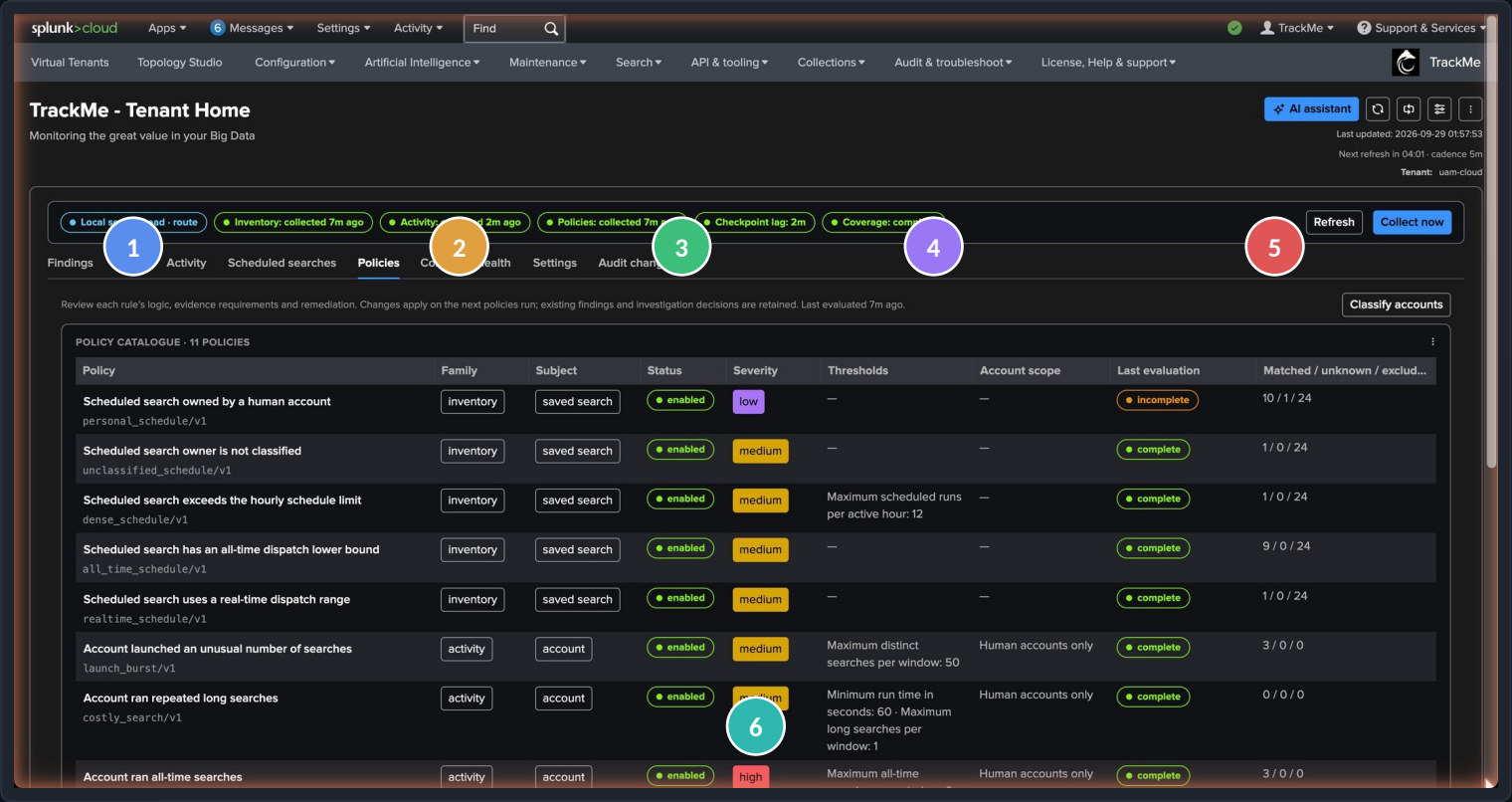
02

PART 3 · TUNE

The policy catalogue and the settings

Eleven policies in three families, six on by default. Severity, thresholds, exclusions and scope per policy; the insight that explains an evaluation; the classification rules and the collection limits.

The Policies tab — the catalogue with its defaults and its counters



- 1 The catalogue**
Eleven policies in three families — inventory, activity, resources — one row each.
- 2 Enabled, opt-in, preview**
Six on by default; four opt-in; the daily baseline deviation in preview mode.
- 3 Severity**
all_time_search is high by default: an all-time search is the one thing that should not happen. all_time_schedule stays medium.
- 4 Thresholds in force**
Compared strictly; the resource policies derive theirs from the measured capacity (auto-thresholds).
- 5 Last evaluation**
Matched, excluded, not applicable, unknown — the counters of the last run, and the entry to the insight.
- 6 Edit and preview**
The modal edits one policy and previews the draft against the current inventory before saving.

The policy insight — why an evaluation matched what it matched



- 1 The verdict**
all_time_search, complete: every applicable account was judged — 3 evaluated, 3 matched, 2 outside the scope.
- 2 Matched subjects**
The accounts behind the findings, with the evidence that matched.
- 3 Not applicable and excluded**
The system accounts — the Splunk context identity, the tenant owner — the policy never judges, listed with the reason.
- 4 The unknowns**
Accounts without a class, classified in place from the insight.
- 5 Seven-day history**
The counters run after run: proof that every classified person is judged, every run.

Eleven policies, three families — six on by default

INVENTORY • the schedules

personal_schedule

on • low

A scheduled search owned by a human account.

all_time_schedule

on • medium

A schedule dispatched with no lower time bound.

realtime_schedule

on • medium

A real-time schedule left running.

unclassified_schedule

opt-in

A schedule whose owner has no class yet.

dense_schedule

opt-in

A schedule denser than the threshold.

ACTIVITY • the audit records

all_time_search

on • high

An account ran an all-time search. High by default: more than zero is a finding.

launch_burst

opt-in

More than N searches launched by an account in a window.

costly_search

opt-in

An individual search above a cost threshold.

RESOURCES • the introspection

resource_consumption

on • medium

An account's CPU seconds over the window against its allowance — 10 % of the search CPU capacity per fifteen minutes.

resource_intensity

on • medium

The count of individually heavy searches — 2 % of the capacity, or 10 % of the smallest host's memory.

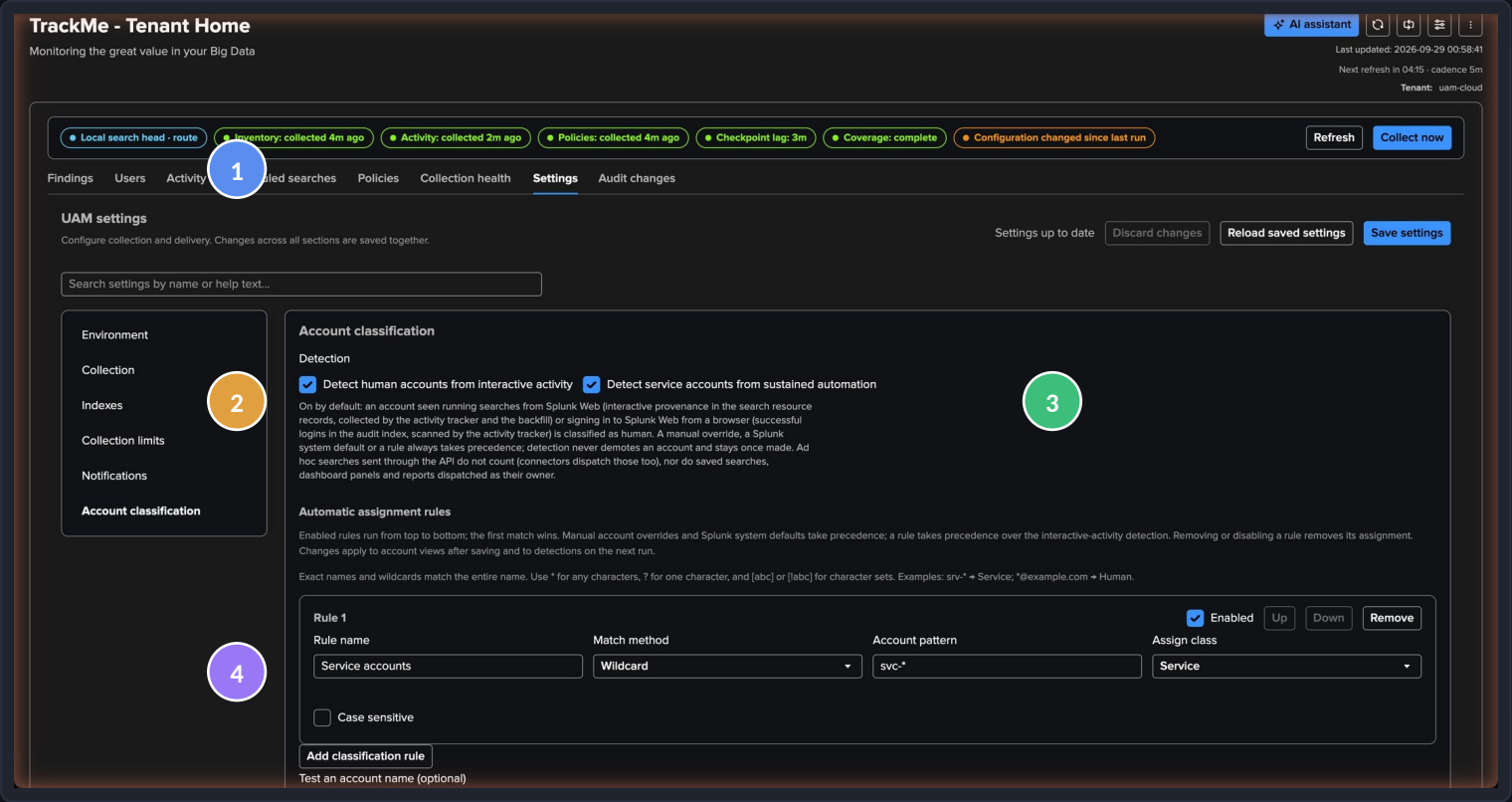
resource_deviation

preview

Daily baseline deviation: records what it would raise after seven fully covered days.

Every threshold is compared strictly; every policy has its own exact exclusions; the trackme and Splunk_SA_CIM apps are excluded everywhere by default. A stored configuration keeps its values across upgrades.

Account classification — manual classes, rules, detection switches



1 Manual classes

Every class set from a row menu, with who and when — svc-trackme and the administrators as system.

2 Rules

Ordered exact or wildcard rules: svc-* → service. Previewed on the known accounts before saving.

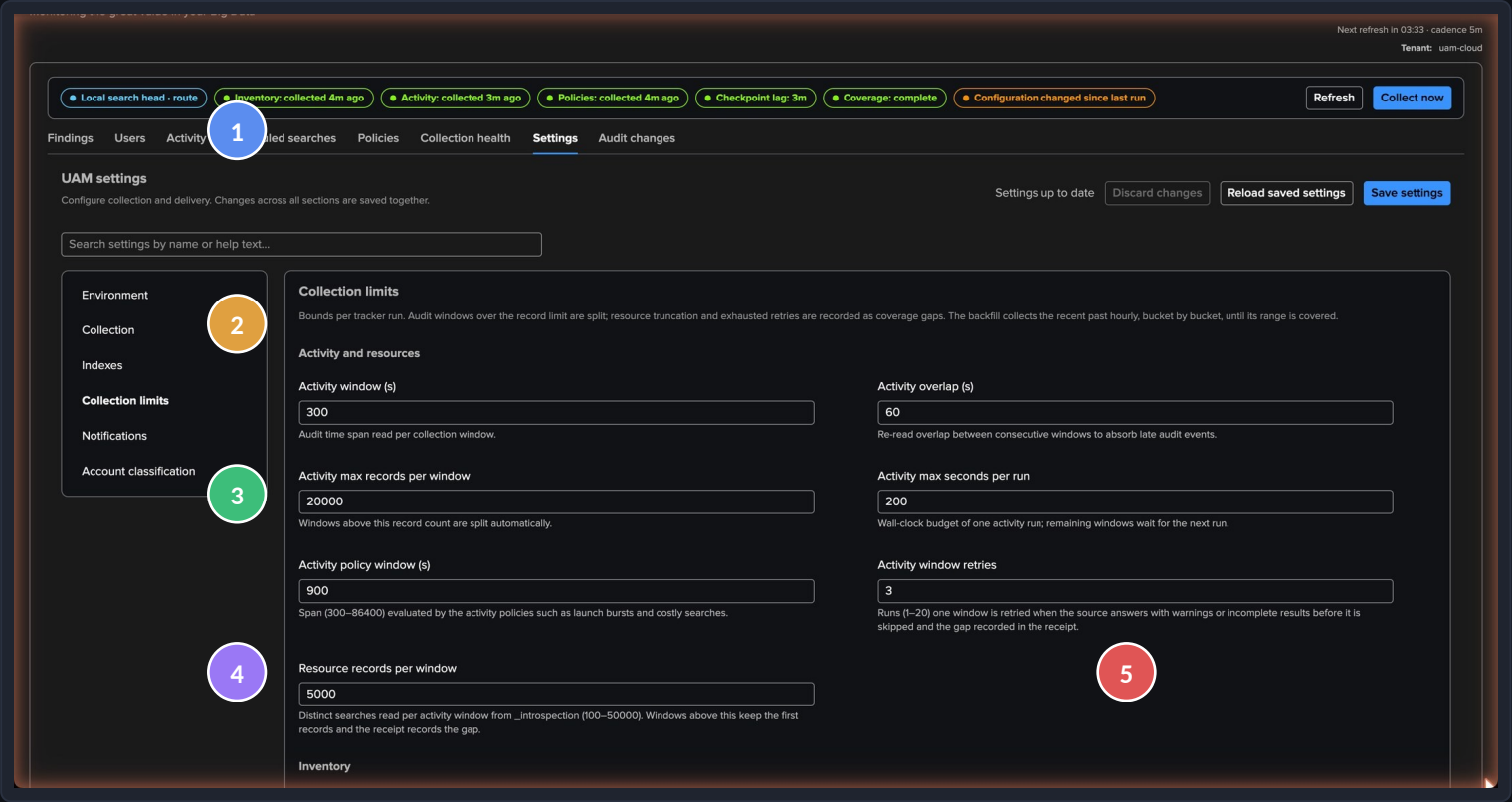
3 Detections

The human and service detections as they stand, with the signal that decided each one.

4 Detection switches

Turn a detection off if a platform convention makes it wrong — a shared browser account, for instance.

Collection limits — every budget is a tunable, not a cap



- 1 **Activity budgets**
Windows and records per run: the audit trail five minutes at a time, a window split when it overflows.
- 2 **Resources and inventory**
The introspection window, the inventory cadences and the listing window — one full listing a day, off-peak.
- 3 **Findings and backfill**
The findings collection cap (50,000 by default), the backfill slot duration and bucket size.
- 4 **Store SPL search — off**
The SPL adds about a third to a record and can carry literals and secrets. Off by default; on when the analysts need the text in the evidence.
- 5 **Validated ranges**
Every budget has a range; the defaults fit a busy cluster.

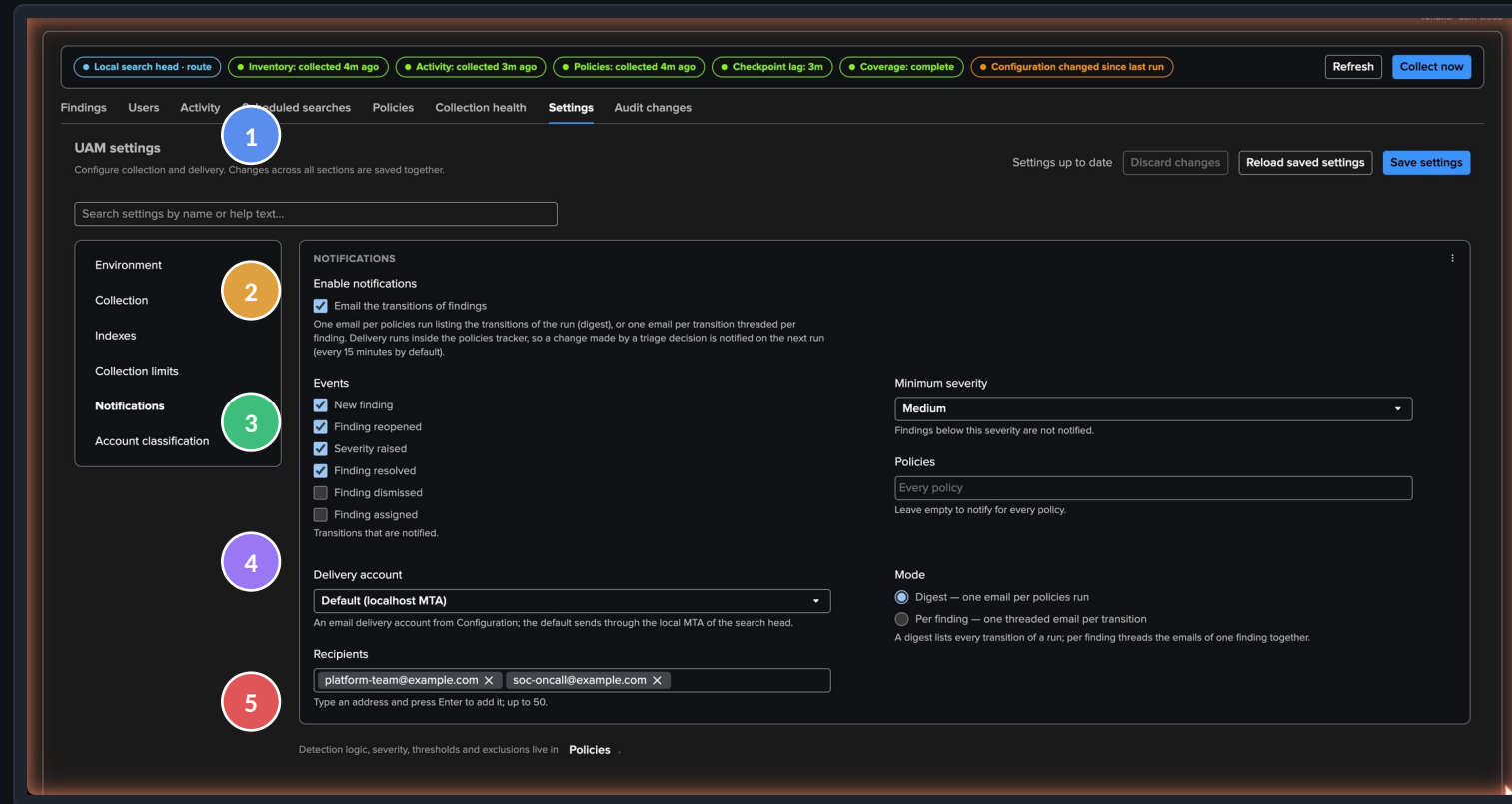
03

PART 4 · OPERATE

Notifications, collection health, the inventory

State-aware emails on the transitions of the findings; the Collection health tab, the recovery of a killed run and the Guardian check; the Scheduled searches tab as the living inventory of the tier.

Notifications — the transitions of the findings, as a digest or as threads



1 Events

new, reopened, escalated and resolved — the defaults; dismissed and assigned stay off.

2 Minimum severity — medium

The low personal-schedule findings stay out of the mailbox.

3 Policies and recipients

Every policy, two recipients — the platform owners.

4 Delivery account

An email delivery account of TrackMe; delivery is fail-open and retried until the transitions expire.

5 Digest or per-finding

Digest: one email per policies run that produced a transition. Per-finding: one email per transition, threaded per finding.

Collection health and recovery — always explainable, self-healing



1 Tracker run view

Every tracker with its last run explained and its seven-day history; the measured capacity; the backfill with its edges and controls.

2 Receipts

What each run read and wrote. A run that dies is replayed, never skipped; a degraded source is retried a bounded number of times, then recorded as a gap.

3 Diagnostics

The reasons of a slow or partial run, in the page rather than in a log file.

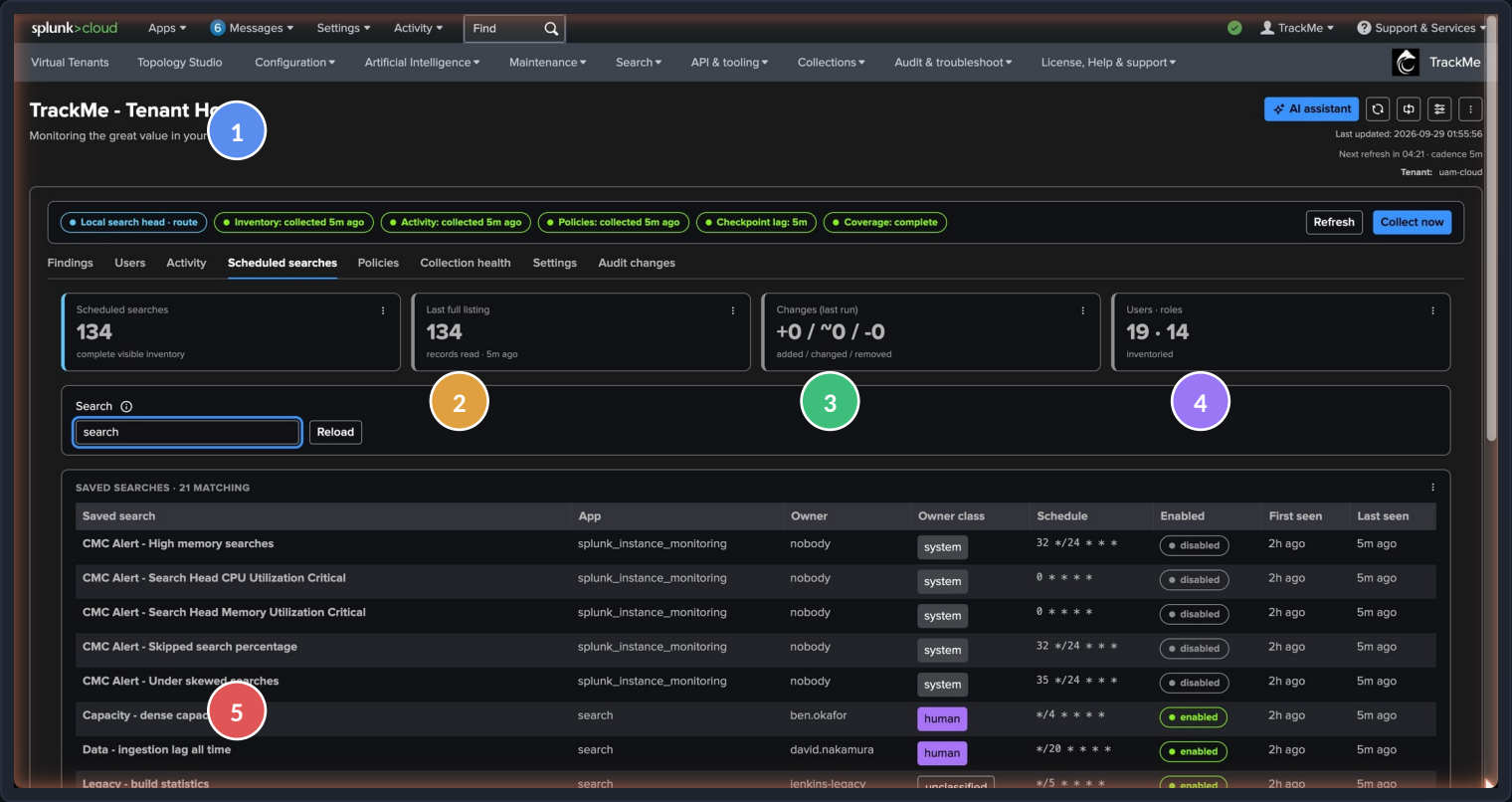
4 Recover stopped run

A run killed mid-way leaves a lock; a lock older than five minutes whose job has stopped is recovered at the next run, or on demand.

5 Guardian — uam_collection_health

A stale or failing tracker, a stalled checkpoint, an overdue listing, a stopped backfill or a failed delivery, surfaced on the Virtual Tenants page. Clears itself.

The Scheduled searches tab — the inventory of the tier, with its owners



- 134 schedules**
Every scheduled search of the tier, read by delta with one full listing a day.
- Owner and owner class**
Who owns it and whether that is a person, a service or the platform — the inventory policies read this column.
- Schedule and dispatch**
The cron expression, the enablement, the earliest and latest time of the dispatch.
- App and tier**
Where it lives; the excluded apps are still listed, they are just not evaluated.
- Every change recorded**
Who created a schedule, when it changed, when it disappeared — trackme:uam:inventory events.

Cost and scale — compact evidence, bounded budgets, a stated Beta

1 record

per search and window — not one line per audit event; the REST access of an account summarised per batch.

Compact

audit records folded per search, REST access summarised per account and batch, one receipt per run — evidence you can afford to keep.

Bounded

five-minute audit windows from a durable checkpoint, introspection window by window, one full listing a day, backfill under an hour per slot.

Licence

the trackme:uam:* evidence counts against the Splunk licence like every TrackMe summary event; Store SPL search off by default.

Beta statement

A complete, supported tenant type. The service detection needs the legacy key=value _audit format and at least seven days of audit retention. The schema of the evidence may move between beta releases; feedback is welcome.

Not in the beta

Concurrent-use and off-hours policies — "N searches running at once", "a human searching at 3 a.m." — need qualified start / end semantics of the search lifecycle. Notifications v2 — delivery to the assignee's own address, a webhook channel. Cross-environment identity linking. Automatic remediation: UAM never edits, disables or dispatches an observed search.

Reads: tstats on indexed dimensions — $O(\text{accounts})$ rather than $O(\text{events})$. State: three KV collections per tenant; the findings collection is capped and accelerated on its time fields.

Where we are

In four parts

● Created

a UAM tenant in six screens — one local tier, a 30-day backfill, a dedicated service-account owner — with five trackers scheduled at once

● Classified

twenty accounts: people detected from Splunk Web, service accounts by one wildcard rule, the platform's own accounts as system

● Tuned

the catalogue: the four opt-in policies switched on with thresholds fitting a small platform, a svc-* rule for the service accounts, the rest left at its defaults

● Operated

with digest notifications above medium, the Collection health tab, self-recovering runs and the uam_collection_health Guardian check

What made it work

Backfill first

Thirty days of history make the people and the services known on day one — instead of a week later.

A catalogue, not a wizard

Eleven policies with defaults that fit a busy cluster; tuning is severity, scope and exclusions, checked in the insight.

Evidence you can afford to keep

One record per search and window, budgets you can read, and a Guardian that says when collection stalls.