



NxSAM API Security — Quick Guide

Quick reference guide for the API Security module

Product release	**NxSAM 2026.09**
Verified against code on	**2026-09-10**
Document revision	**b441b08-dirty**
Audience	**End users**
Document owner	**eSec Forte Technologies — Product Documentation**

Screenshots are taken from a demonstration data set. Figures are illustrative.

eSec Forte Technologies

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Table of Contents

API Security

- Overview
- Reading the API Security screens
- API Security: every screen, column and action
- API Security: find the right screen

Reference

- Glossary

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

API Security

API Security

API Security tracks weaknesses in service endpoints — the interfaces your applications, partners and mobile apps call. APIs are assessed separately from the applications that use them, because they are separately exposed and separately exploitable.

Why APIs get their own module

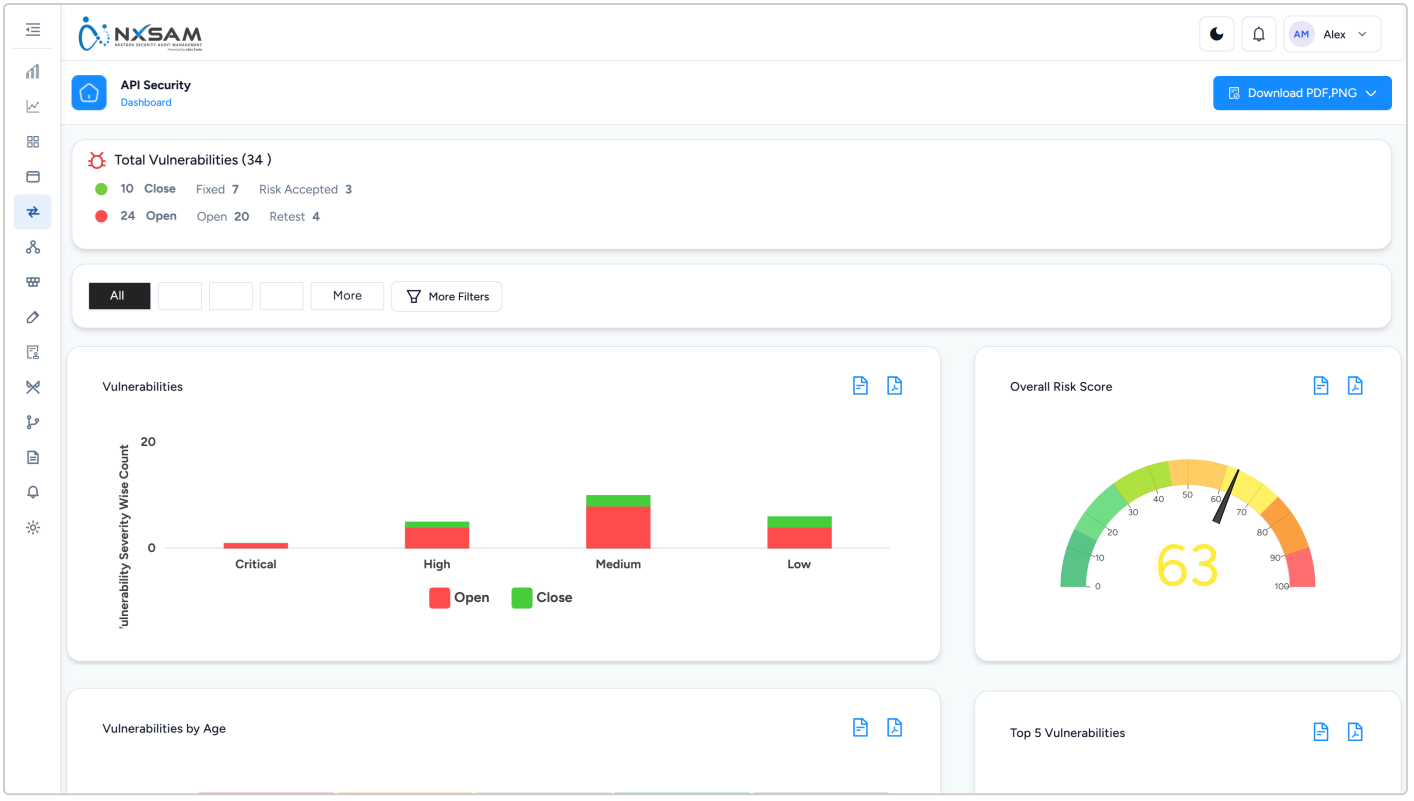
An API is often reachable by more parties than the application in front of it, and it rarely has the protections a user interface does. The characteristic API weaknesses are also different in kind:

- **Authorisation flaws** — a caller can request another customer's record simply by changing an identifier. The most common and most damaging API weakness.
- **Excessive data exposure** — the response contains more than the caller needs, and the filtering happens in the client.
- **Missing rate limits** — an endpoint that can be called without restriction.
- **Forgotten versions** — an old API version left reachable after its replacement shipped.

None of these are visible from the user interface of the application using the API, which is why the API is assessed in its own right.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

How the module is organised



The API Security dashboard.

Screen	Shows
Dashboard	API posture at a glance
Audits	Assessments performed against your APIs
Vulnerabilities	The findings from those assessments

The lifecycle is the same as everywhere else in NxSAM: findings are discovered, approved, given an SLA by severity, remediated, then validated to close.

Identifying the asset

API assets are identified by their endpoint. Versioned APIs are normally separate assets — `/payments/v1` and `/payments/v2` have different code, different findings and often different owners.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

This matters for the forgotten-version problem: if the old version is not registered as an asset, nothing assesses it, and nothing reports that it is still reachable.

Working with API findings

- **Authorisation findings first.** They are usually exploitable immediately, need no special access, and expose other customers' data.
- **Check whether a finding is systemic.** One endpoint missing an authorisation check often means a whole class of endpoints shares the pattern. Fixing the pattern is worth more than fixing the instance.
- **Retest after fixing.** Authorisation fixes are easy to get subtly wrong, and a fix that works for the tested case may not hold for adjacent ones.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Reading the API Security screens

API Security mirrors Application Security closely — same two-status model, same audit structure — with a handful of differences that matter when tracing a finding.

What is different from AppSec

	Application Security	API Security
Asset	An application	An API asset — a collection of endpoints
Identified by	URL or package	Base URL, usually versioned
Finding attached to	The application	The API asset, and often a specific endpoint
Status column heading	**Current Status**	**Current Status**, under a different underlying name

The last row is the practical one. The two modules hold the same concept under different names beneath the surface, so a report or export template built for one does not carry the other's status column — which is the usual reason an API export appears to be missing its status. Build the template against the module you intend to export.

The columns

Column	Notes
Track ID	Quote in tickets; survives re-testing
Issue title	The finding, with its category

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Column	Notes
Risk	Severity, derived from the score
CVSS score	The raw score before context
Affected URL	The endpoint, not just the asset — the most useful column here
Validation status	Whether the finding is agreed
Client vuln status	What is being done about it
Discovered on	Last seen, not first
SLA duration / expiry	The window and the deadline
Creator	Who raised it — often a tester rather than a tool
Risk acceptance comment / date	Present inline where the finding was accepted

Affected URL is where API findings differ in practice

An application finding says "this app has XSS". An API finding says "this endpoint accepts an unauthorised object reference" — and the endpoint is the actionable part.

Two consequences:

- **The same weakness on two endpoints is two findings**, correctly. Fixing one does not close the other, and a shared fix in middleware closes both only when the next assessment confirms it.
- **Grouping by affected URL** is usually more useful than grouping by severity, because it maps to the code that has to change.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Why API findings are often manual

Automated scanners struggle with APIs: they need a specification, valid authentication and realistic payloads to reach anything interesting. Much of what matters — broken object level authorisation, excessive data exposure, mass assignment — requires understanding what the endpoint is *for*.

So a higher proportion of API findings are raised by a tester than by a tool, which is why the **Creator** column is worth reading and why evidence attached to the finding carries more weight here than in a scanner-driven module.

Audits scope the list

Like AppSec, this screen always shows findings within an audit. A finding you expect and cannot see is usually in a different audit rather than missing — check the audit selector before treating it as absent.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

API Security: every screen, column and action

API Security mirrors Application Security in structure and differs in several details that matter when tracing a finding — including two audit-table implementations whose menus are worded differently for the same operations.

The audit list

Column	What it holds
Request Info	Who requested the audit, when, and the request ID
Audit Details	Audit name and the API asset it runs against
Vulnerabilities	Open counts with the severity split and the risk score
Audit Properties	The chips — type, cycle, category, scope, considered
Audit Status	A dropdown offering only the legal next steps

Row actions — twelve

Action	What it does
View Vulnerabilities	The findings for this audit
Vulnerability Overview	Severity and asset summary without leaving the list
Manage Audit Scope	Records what was in and out of scope

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Action	What it does
Manage Test Summary	A free-text description of what testing was performed
Assign Auditors	Who performs the work
Edit Audit Details	Name, cycle, dates, properties
Edit Tags	Labels for grouping and reporting
Update Audit Start Date	Changes the start date after creation
Re-Audit	Creates a follow-up audit against the same scope
View Audit Chain	The original and every re-audit in sequence
View Audit Status History	Every status change with who made it
Delete Audit	Soft delete

Manage Test Summary is what makes an API audit defensible

APIs are largely tested by hand, so the summary is the record of *what was actually attempted* — which endpoints, which authentication, which abuse cases. An audit without one cannot be reviewed later, and an auditor asking "what did you test" is asking for this field.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

⚠ Two audit tables, two vocabularies

The module ships two audit-table components. One labels its menu **Assign Auditors**, **Edit Audit Details**, **Manage Audit Scope**; the other labels the same operations **Assign Api Auditors**, **Edit Api Audit Details**, **Manage Api Audit Scope**, and **View Api Audit Chain** / **View Api Audit Status History**.

They do the same things. If a caller quotes a menu item you cannot find, this is why — ask which screen they reached it from rather than assuming a version difference.

The vulnerability list

Column	Notes
Vulnerability	The finding and its category
Tool	Which scanner reported it, or Manual
Risk	Severity as derived from the score
Validation	Whether the finding is agreed
Current Status	What is being done about it. API Security keeps its own status list, separate from the AppSec one
Discovered On	Last seen, and blank until the finding is approved
SLA Details	Duration and expiry
Artifacts	Evidence attached to the finding

Row actions — eight

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Action	What it does
Edit Vulnerability	The finding's own fields
Edit CVSS Metrics	The CVSS vector, component by component
View CVSS Metrics	The same, read-only
Edit Issue Master Details	Edits the underlying catalogue entry, not just this instance
Edit Sequence	Reorders findings within the report
View Vulnerability History	Every change to the finding
View Client Status History	Status changes specifically, with who made them
Delete Vulnerability	Removes the finding from the audit

⚠ Edit Issue Master Details changes the catalogue, not the finding

The two edit actions sit next to each other and do different things. **Edit Vulnerability** changes this occurrence. **Edit Issue Master Details** changes the master entry — the title, description and remediation that *every* occurrence of that vulnerability inherits, across every audit and every asset.

Correcting a typo through the master is right. Recording something specific to one endpoint through it is not, and the effect is not obvious from the dialog.

Edit Sequence matters because API findings are usually presented as a narrative — a chain where one weakness enables the next. The sequence is the order they appear in the generated report, and it is set deliberately rather than by severity.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Artifacts

Evidence attached to a finding, structured as steps:

Column	What it holds
Step	The position in the reproduction sequence
Description	What was done at this step
Images	Screenshots for that step

This is the reproduction path. An API finding is often only demonstrable as a sequence — authenticate as A, obtain a token, alter an identifier, read B's data — and a single screenshot proves nothing. The step structure is what makes it re-testable by someone other than the original tester.

Vulnerability history

Column	What it holds
Audit ID / Audit Details	Which audit the change belongs to
Risk	The severity at that point
Vulnerability Status	The status it moved to
Created On	When
Status Changed By	Who

Because it spans audits, this is how you see that a finding was closed in one assessment and re-opened by the next — the single most common dispute on a re-test.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Add Old Vulnerabilities

Opened when creating a re-audit. It lists findings from previous audits — `vId`, **Vulnerability**, **Audit Name**, **Risk**, **Client Status** — so the ones still outstanding can be carried into the new audit rather than re-entered.

Carrying a finding forward preserves its history and its original discovery date. Entering it again creates a second finding with neither, and the two then age separately, which is why the option exists.

Edit Tags

Tag ID and **Tag Name**, multi-select. Tags are shared across audits, so renaming one here renames it everywhere it is applied.

The status models

API Security keeps its own master tables. The values overlap Application Security's without matching them, so a status list quoted from one module will not fit the other.

Audit status — fourteen values

In sequence: `Saved`, `Reverted`, `Requested`, `Approved`, `Scheduled`, `Work In Progress`, `Not Hold`, `Completed`, `Report Published`, `Aborted`, `Review Requested`, `Processed`, `Changes Requested`, `PMO Verified`.

- **Closed types** — `Completed` and `Aborted` only.
- **Comment required** — `Aborted`, `Changes Requested` and `PMO Verified`, and any other status the administrator has marked as needing a reason. Moving an approved audit to **Hold** asks for one.
- `Report Published` sits after `Completed`: the assessment is finished and the report has been issued, which are separately meaningful to a customer.
- `Not Hold` is the release from a hold rather than a hold itself, despite reading like one. Nothing is placed *into* `Not Hold` to pause it.

Validation status — five values

`Saved`, `Submitted`, `Approved`, `Closed`, `False Positive` — the same five as Application Security, and notably **not** the three Network PT uses.

Client status — fourteen values

`New`, `Patch in progress`, `Patched`, `Not Patched`, `To be patched`, `On hold`, `Reopen`, `Validated`, `Not Applicable`, `Risk accepted`, `Risk acceptance`, `Duplicate`, `Exception Handled`, `Expired`.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Closed types are `Validated` and `Exception Handled`. `Not Applicable` and `Risk accepted` require a comment; `Risk accepted`, `Risk acceptance` and `Expired` display a date alongside the status.

The dashboard

Each widget exports individually through **Download PDF** and **Download PNG**.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

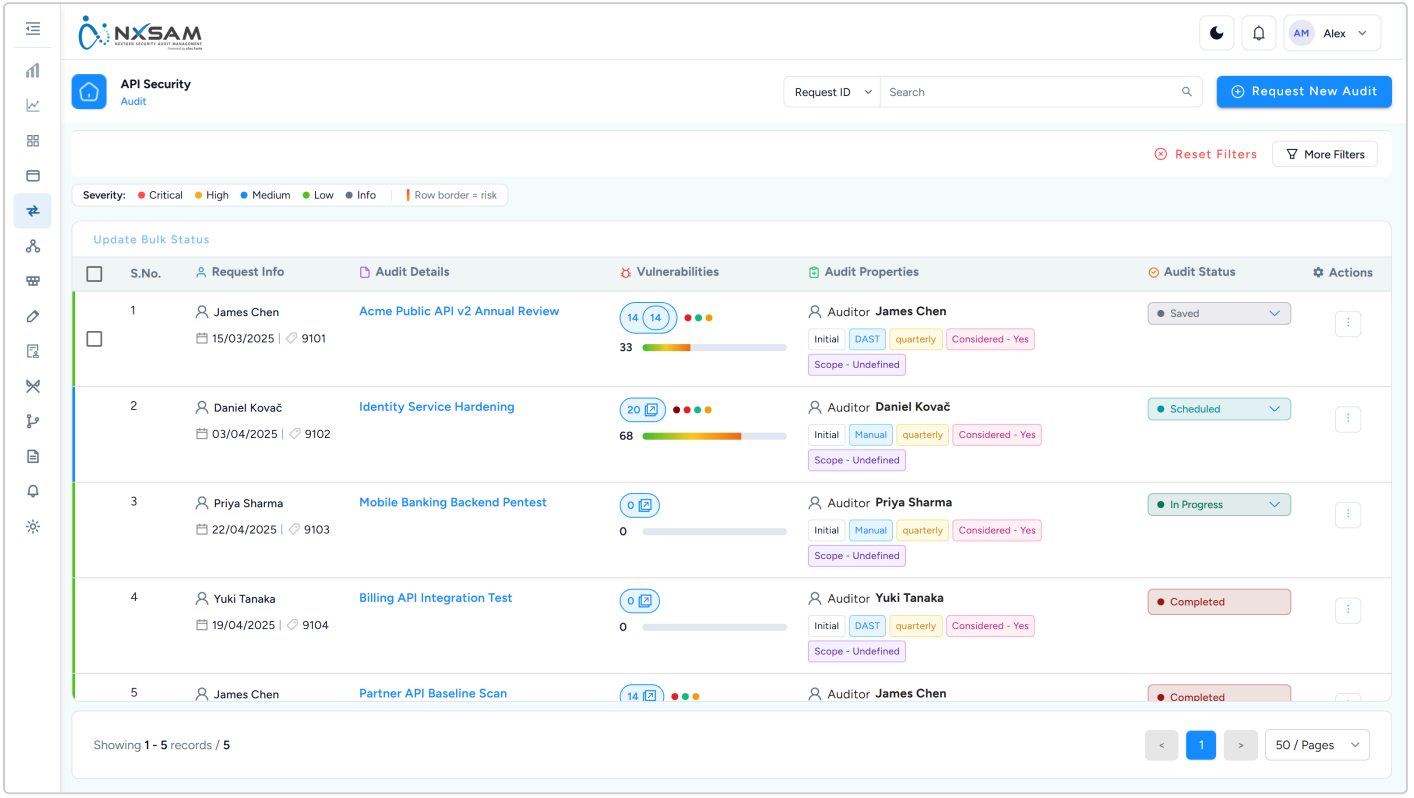
API Security: find the right screen

Every screen in this module, what each one is for, and where it lives. Use it to find the right screen quickly, and to confirm you are on the one a colleague means — several screens in this product have similar names and different jobs.

Screenshots come from a demonstration data set, so the figures are illustrative. Click any screenshot to open it at full resolution.

API Audits

Assessments against API assets. Reached the same way as AppSec audits, tracked separately.

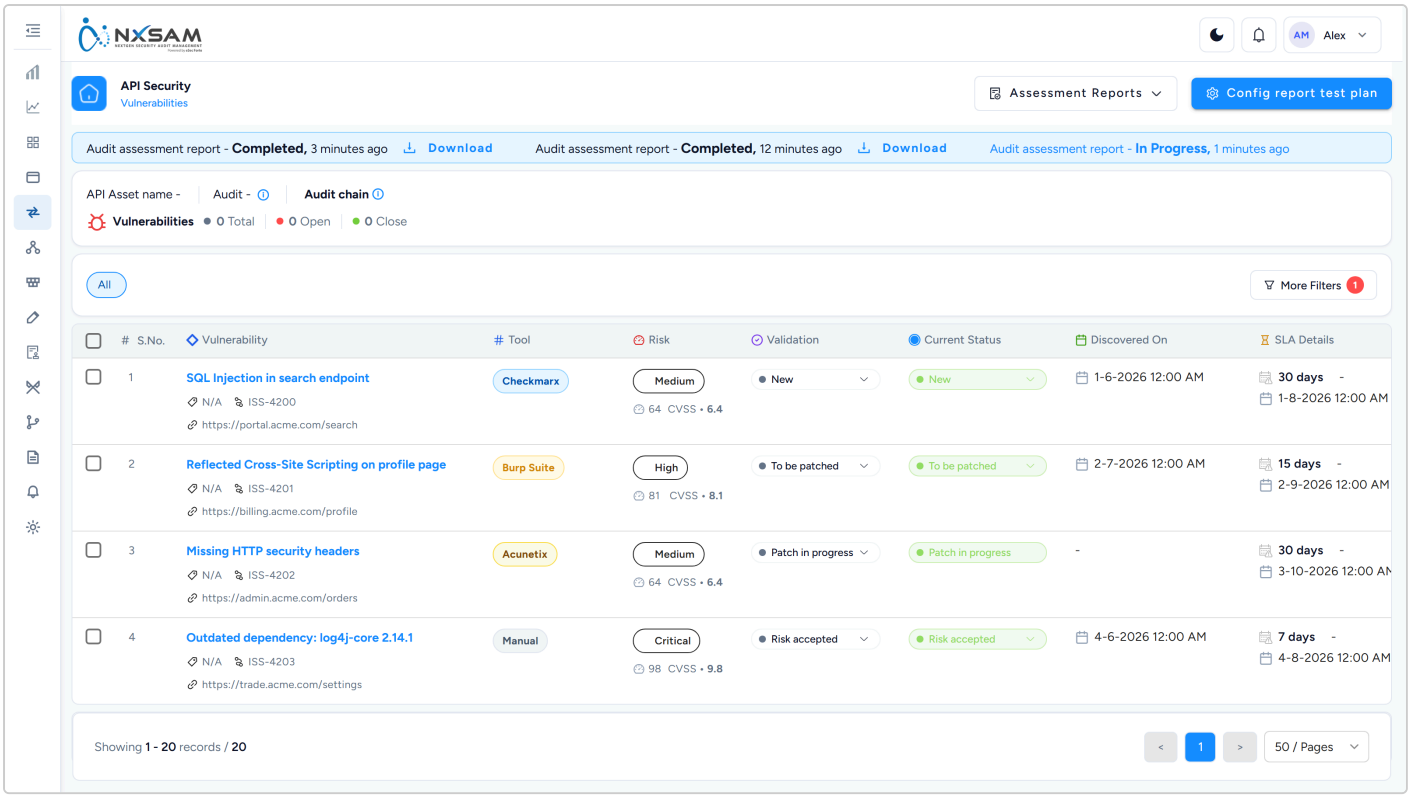


Api Security → Audits

API Vulnerabilities

Findings within an API audit. Same two-status model as AppSec — validation and client status move independently.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.



Api Security → API Vulnerabilities

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Reference

Glossary

Terms as NxSAM uses them. Several are ordinary security words with a narrower meaning here, and those are the ones worth reading even if the term is familiar.

Assets and scope

Asset — anything that can hold a vulnerability: a web or mobile application, an API, a network host, a thick client, or a container image. Each type is a separate inventory with its own columns and its own licensed limit.

Asset property — an attribute recorded against an asset, such as criticality or whether it is internet-facing. Properties are more than description: they narrow SLA durations and feed the contextual risk score.

Bifurcation — splitting one application into separately assessed parts, so a large system can be tested module by module rather than as one unit.

Collection — a grouping of container images that comes from the scanning platform rather than from NxSAM, and is not editable here.

Scope — what an assessment covers. For applications this is the application; for compliance it is a host list, which is why compliance scope disputes are settled with an exported list of addresses.

Assessments

Audit — one assessment of one asset. It has a type, a category, a cycle, a scope, and a status of its own, separate from any finding's status.

Audit cycle — which round of a recurring programme an audit belongs to.

Confirmatory audit — a re-test created against a parent audit to verify that fixes hold. It is evidence about the original rather than an independent assessment, so the two should not be counted as two rounds of testing.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

Considered — the marker deciding whether an audit contributes to aggregate figures. Normally the latest audit per asset that reached a final state. An audit that is *not* considered still shows its findings on the module screen while contributing to no dashboard or report.

Initial audit — a first assessment, as opposed to a confirmatory one.

Findings

Vulnerability (also *finding*, *issue*) — one weakness on one asset, found by one assessment.

Vulnerability master — the catalogue entry (title, description, remediation) inherited by every occurrence of that vulnerability everywhere. Editing the master changes all of them.

Posture — Policy Compliance's unit: one control checked against one host. Every check produces a posture whether it passed or failed, which is why this module's volumes dwarf the others.

Artifact — the evidence attached to a finding: the steps, description and images that let someone else reproduce it.

Instance — which occurrence, where the same control or finding applies more than once to the same asset.

Status

Validation status — whether a finding has been confirmed. Nothing is counted anywhere until this reaches *Approved*, which is also when **Discovered On** is populated.

Client status (also *current status*) — what is being done about the finding. Of its values, only two close it.

Closed type — the flag deciding whether a status stops something counting as open. It is configuration rather than a fixed rule, so a deployment can change what "closed" means.

Posture status — what a compliance check returned: pass, fail or **error**. Error means the check could not run — a scanner or credentials problem rather than a compliance result.

Sub status — the qualifier beneath a main status: which kind of acceptance, or which stage of an approval.

Time and SLA

SLA duration — how long a finding of a given severity on a given asset type may remain open. Configurable, and narrowable by asset property.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

SLA expiry — the resulting deadline. Changing an SLA duration re-derives expiry dates, which can make a batch of findings overdue with no data change at all.

Discovered On — when a finding was first observed. Blank until the finding is approved.

Last observed — the most recent sighting, which is how a finding is known still to exist.

Overdue — past SLA expiry and not closed.

Decisions

Deviation (also *risk acceptance*, *RA*) — a recorded decision not to fix something, with an approval chain and an expiry. It does **not** close the finding: the exposure is still live and still counts as open.

Exception — a status marking an agreed exception rather than a resolution. Reporting that conflates the two overstates remediation.

Auto reject end date — when a temporary acceptance lapses on its own.

Integrations and processing

Integration — a connection to a scanning tool, either pulling from its API or parsing an uploaded report.

Vulnerability key — the columns deciding whether an incoming finding is one already seen. Changing it redefines "the same finding" and splits history.

Queue — asynchronous work: imports, syncs, report generation. A screen that seems not to have updated usually has a queue entry that explains it.

Data process — how far a tool-fed audit has got through the import pipeline; the difference between "the upload finished" and "the data is ready".

Beat — a scheduled job maintaining derived data, such as recalculating SLA states as dates pass. A stopped beat produces figures wrong by a fixed interval and raises no error.

Governance

RLCM — Red Line Compliance Management: assessment of controls against frameworks, rather than of systems for weaknesses.

TPRM — Third-Party Risk Management: the same control assessment applied to suppliers. It shares screens with RLCM and differs by the audit's type.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.

UCF — Unified Control Framework: the mapping that lets one control satisfy several frameworks at once, so answering it once covers each of them.

Control — a requirement being assessed, as opposed to a weakness being found.

Standard — the framework a control belongs to, such as ISO/IEC 27001 or PCI DSS.

Reporting

Considered audit filter — the switch on Report Search that reproduces either the dashboard's numbers or the module screens'. Comparing the two settles most disagreements about counts.

Design template — the layout a report renders through, separate from the template that decides its content.

Share view against share copy — sharing a template as a view keeps one definition that your edits reach; sharing a copy creates an independent template that will drift.

Copyright 2010-2026 eSec Forte® Technologies Private Ltd.