PULSE

Pulse Documentation

Pulse · Mobile App · Appliance

Revision 2026-09-09

docs.pulsehq.de

Contents

9 Appliance

1 What's New

Notable, user-facing changes to Pulse — new features, improvements, and fixes you'll actually notice. Newest first. Follow the links for full details in the feature documentation.

September 9, 2026

Fixed

- **Condition preview evaluates against the correct baseline at each point in time** — When previewing a condition over a historical time range, the evaluation now uses the baseline version that was actually in force at each moment; a condition with multiple baseline revisions no longer returns backtest results that would differ from what the live monitor decided at that time. See Monitors.
- **Bulk-deleting monitors lists blockers in the order you selected them** — When a bulk monitor deletion is refused because some monitors have attached conditions, the dialog now names those monitors in the same order they appear in the list, matching the order you selected them.

September 8, 2026

New

- **Filter monitors by data source** — The monitor list now has a Data source filter so you can show every monitor reading a specific device at once, making it easy to select all and mute or act in bulk during a maintenance window. See Monitors.
- **Scope incident reports to specific data sources** — Report configurations now accept a data-source filter; generated reports then cover only incidents raised by monitors that read those devices. See Reports.
- **Delete several conditions at once** — The conditions list has the monitor list's selection mode: **Select** turns every row into a checkbox, select-all and Shift+click pick a run, and the bar at the bottom deletes the lot behind a typed confirmation. A selection holding a condition a monitor alerts on or an objective promises about is refused as a whole, and the dialog then names what is holding it and offers to delete the rest. See Monitors.
- **Bulk deleting monitors says how many conditions went with them** — Deleting the last monitor that alerts on a condition has always deleted that condition too; now the confirmation toast reports both numbers, so clearing out monitors after an upgrade no longer leaves you hunting for graphs that are already gone. See Monitors.

Improved

- **On the appliance, a rejected update file now says what is wrong and what to do** — When an update bundle failed its check, the screen showed only the checking tool's own message ("verification failed: checked against wrong key"). It now leads with one sentence saying what was found — that the bundle was signed for a different appliance, copied incompletely, or altered after signing — followed by the next step. The technical message stays below it, dimmed, for support, and long messages now wrap on narrow consoles instead of running off the edge.

Fixed

- **Muted heartbeat monitors now reflect their true health state** — A muted heartbeat monitor previously stopped being checked and stayed frozen in its last-known state even when the heartbeat stopped arriving; it is now evaluated continuously and will turn unhealthy when overdue, so the status indicator stays accurate without the monitor raising an incident.
- **Condition monitors used the wrong sampling interval for freshness checks** — Monitor freshness was calculated from a field on the data point record instead of the data source, which could cause the monitor to treat on-time samples as stale. The correct data-source interval is now used.
- **AI Canvas and MCP monitor listing silently ignored type filters** — A mismatched argument order caused type filters to be dropped and other arguments to shift position, returning incorrect monitor lists; the tool now applies all filters correctly.

Removed

- **Legacy heartbeat monitor edit page** — The `/monitors/heartbeat/:id` URL now redirects to the monitor list; heartbeat monitor settings are managed from the data source they belong to.

September 6, 2026

Fixed

- **Tag filter rows stay in place while selecting** — In the monitor list's tag filter, rows used to reshuffle as you ticked boxes because the list re-sorted against live match counts; they now hold their position for the entire time the picker is open and only re-rank when you close and reopen it. See Monitors.
- **Occurrence history on upgraded monitors restores automatically** — After upgrading an appliance where node, endpoint or heartbeat monitors were converted to condition monitors, Pulse now migrates the remaining resolved incident history in

the background at a steady pace, so the full occurrence list gradually becomes visible without any extra action.

September 5, 2026

New

- **AI Canvas and MCP can inspect condition graphs and incident timelines** — Two new MCP tools let your AI assistant go deeper into investigations: `get_condition` reads the full condition graph a monitor alerts on and shows what every node currently evaluates to and when that value began — answering which input triggered an alert; `list_incident_events` retrieves an incident's complete timeline: monitor failure and recovery decisions, every notification delivery and to whom, acknowledgements, escalations, and resolutions. See AI Canvas and Organization Settings.

Fixed

- **On the appliance, crash-looping services are now detected as unhealthy after an update** — If a service was broken by an update but configured to restart automatically, a single snapshot of failed units would miss it — the service would cycle endlessly through "activating" and the update report would say "Update successful" despite the broken daemon. The appliance now samples systemd's state repeatedly over a short window and counts a service as failing if it is caught in a crash loop during that time, ensuring the post-update health report reflects services that are actually broken.

September 4, 2026

New

- **MCP clients now connect via OAuth** — Instead of manually creating access tokens, AI assistants and tools connect to Pulse's MCP endpoint via browser-based OAuth: add the endpoint URL (shown in Organization Settings → MCP Clients) to your client such as Claude Code, then approve access in the consent screen that opens in your browser. Organization admins can see every authorized client, who authorized it, and when it was last used, and can revoke access at any time.
- **AI Canvas and MCP can list data sources** — The AI assistant can now list your organization's data sources and their data points, making it possible to ask which sources are configured and which data points have no monitors watching them. See AI Canvas.

Fixed

- **Upgrading an appliance with a long incident history now completes** — On appliances carrying months of incidents, the upgrade that converts node, endpoint and heartbeat monitors into condition monitors could run for over an hour and be rolled back before it finished, leaving the appliance on its old version. The upgrade now rewrites only the incidents that still need their new key — open ones, and resolved ones still inside their monitor's reopen window — instead of the entire history, and the appliance allows up to an hour for the database upgrade before intervening. Plan a maintenance window of up to an hour for this upgrade if your appliance has a long incident history.
- **MCP data graphs no longer invent readings between samples** — In MCP responses, ASCII sparkline charts previously stretched sparse series to fill the full plot width by interpolating between actual samples; they now render at the series' true resolution so a sparse dataset is shown narrower rather than filled with fabricated values.

September 2, 2026

New

- **Name and describe your data points** — You can now give any data point a human-readable name and description; the name appears throughout the app wherever the data point is shown, making it easy to identify readings at a glance.

Improved

- **Monitor count on data-point rows** — On a data source's detail page, the monitor column now shows a single "X monitors" button that opens a searchable picker instead of individual links per monitor, keeping the page tidy for data points with many monitors.
- **Inline hints in the condition editor** — "Valid for" and window duration fields in the condition editor now have icons that explain the field's purpose and accepted format, visible without leaving the editor. See Monitors.
- **AND/OR gate hints** — AND (All) and OR (Any) gate nodes in the condition editor now display an that describes their logic, making the difference between the two types immediately clear. See Monitors.
- **Sparse data notice in condition preview** — The condition preview now flags when a signal has samples for only part of the selected range, so a limited backtest result is clearly labelled rather than mistaken for a full one. See Monitors.
- **Tag badges show the full label on hover** — When a tag label is truncated in a list row, hovering the badge now reveals the complete label in a tooltip.

Fixed

- **Interface labels follow the selected language** — The "Close" button on dialogs and side panels, and the sidebar toggle button, now announce in the user's active language rather than always using English.
- **Filter chips hide counts on paginated lists** — On the Data Sources page, filter chips no longer show misleading counts when the full result set spans more than one page; the chip shows only the label in those cases.

September 1, 2026

New

- **Monitors panel on data source pages** — Data source detail pages now show a "Monitors" section listing every monitor that alerts on the source's data points, with a link to open each one directly.
- **Approximate history for new objectives** — When you create an objective, Pulse now backfills an approximate compliance history over the window before creation and labels it accordingly, so you see a meaningful picture immediately instead of "not enough data" for most of the first window. See Objectives.
- **Activate-on-import toggle in monitor CSV import** — The CSV import wizard now has a toggle to choose whether imported monitors are activated immediately after the import or left paused for review in the monitor list. See Monitor CSV Import.

Improved

- **Escalate an incident to an on-call schedule** — The manual "Escalate..." dialog now offers on-call schedules alongside users and teams, the same recipients an escalation policy step takes. The schedule pages whoever is on call at that moment, and the timeline entry names the schedule you picked. See Incidents.
- **Duration column accepts human-readable values** — In the monitor CSV import, the duration field now accepts values like "5m" or "30s" instead of requiring a raw number of seconds. See Monitor CSV Import.

Fixed

- **OPC UA browse retry after failure** — When the "Add data point" browse dialog fails to load nodes from an OPC UA device, a retry button now appears so you can try again without closing the dialog.
- **OPC UA browse dialog shows the full node set** — The OPC UA node browser previously omitted some nodes declared on the source; all declared nodes now appear correctly in the browse dialog.

- **Monitor CSV import validates threshold conditions** — The importer now validates generated threshold conditions before writing them, preventing monitors with invalid parameters (such as a negative duration) from being created silently. See Monitor CSV Import.

August 31, 2026

New

- **Operating hours gate in escalation policies** — Escalation policies now support an "Operating hours" step that holds the escalation at that point until the configured hours apply; while outside those hours the policy waits, and when the window opens again it continues automatically. The pause is logged in the incident timeline. See Escalation Policies.
- **Invert operating hours** — Both your personal notification preferences and operating-hours gate steps now have an "Invert" toggle: when enabled, the window applies *outside* the configured hours instead of during them, useful for on-call coverage that spans nights and weekends. See Escalation Policies.
- **Filter by escalation policy** — The monitor, incident, objective, and condition lists now include an escalation-policy filter to narrow results to policies you care about. See Monitors and Incidents.
- **Right-click context menu on incident rows** — On desktop, right-clicking an incident row opens a context menu with the same actions as the row's menu button: acknowledge, resolve, escalate, and open.

Improved

- **Acknowledgement push to other responders** — When someone acknowledges an incident, the other users still being paged receive a push notification so the whole response team knows the incident has been picked up. See Incidents.
- **Report entities link to their live pages** — Monitor names and other entities in the report viewer are now clickable links that open the corresponding live page in the app. See Reports.
- **Data source rows are now fully clickable** — Clicking anywhere on a data source row navigates to its detail page, consistent with monitor and incident rows.
- **App stays navigable when a page crashes** — If a page throws an unexpected error, the sidebar and navigation remain accessible and a clear error message appears in place of the crashed page, instead of a blank screen that requires a full reload.

Fixed

- **Baseline monitors now match targets correctly** — A bug causing nightly baselines to key on retired monitor subtypes rather than adopted data points is fixed; baseline monitors that previously showed "insufficient data" in error should now compute correctly. See Monitors.
- **Report snapshots with missing fields now open correctly** — Reports generated before the objectives feature was added now render as expected instead of failing, and a report section that fails to load now shows an isolated error rather than taking the rest of the report with it. See Reports.

August 30, 2026

New

- **Heartbeat grace period** — Heartbeat data sources now have a configurable "Grace period" that sets how late a ping may arrive before Pulse raises an incident, covering normal job-to-job variation in runtime; the data source detail page shows the expected interval and grace period together as the alerting deadline. See Heartbeat.
- **Open an objective by clicking its row** — Clicking anywhere on an objective's card in the Objectives list now navigates to that objective's detail page, consistent with how monitor and incident rows work. See Objectives.
- **"Open condition" in the objective row menu** — The actions menu on each objective row now includes an "Open condition" item that navigates directly to the condition in the editor. See Objectives.
- **Prometheus series as condition sensor nodes** — Prometheus series can now be used directly as Sensor nodes in the condition editor, letting you alert on Prometheus metrics with the same graph-based conditions as any other data source. See Monitors and Prometheus.
- **Push notifications carry an icon for what happened** — Each incident push now starts with an icon, so a glance at the lock screen says which of them it is: □ a new incident, □ someone acknowledged, □ resolved, and □ a colleague escalated the incident to you personally. See Mobile notifications.

Improved

- **Condition editor adds a "Summary" tab** — The textual expression of a condition is now behind a dedicated "Summary" tab in the preview panel (alongside "Preview" and "Live"), so the readable form of the graph no longer competes for space with the backtest or live values. See Monitors.
- **Condition save errors now identify the problem** — When the server refuses to save a condition, the specific blocking issues are counted in the toolbar chip, highlighted on the affected nodes, and described beside the Save button, so you

know exactly what to fix rather than seeing only "The condition can't be saved yet." See Monitors.

- **Objective delete dialog explains what happens to the condition** — The confirmation when deleting an objective now says whether the condition it was about will be deleted with it (if this objective was its only consumer) or kept intact (if a monitor or another objective still uses it). See Objectives.
- **Report burndown charts start folded** — Each objective's error budget burndown chart in the report viewer now starts collapsed with a "Show budget history" button, keeping the reliability section concise until you expand it. See Reports.

August 29, 2026

New

- **Manually escalate an incident to any user or team** — An "Escalate..." action on open incidents lets you page specific people directly, choosing which channels to use (email, SMS, call, or push). The escalation is additive to the automatic escalation policy and works on both unacknowledged and acknowledged incidents; once resolved the action is no longer offered. Each manual escalation appears as its own entry in the incident timeline. See Incidents.

Improved

- **Reports collapse the incident list by default** — The per-incident detail section in a report viewer now starts collapsed, showing only the incident count on the toggle, so the report summary is immediately readable without scrolling past potentially thousands of rows. Expand it when you need the full list. See Reports.

August 28, 2026

New

- **Dedicated create and edit page for condition monitors** — Creating a condition monitor now has its own page at **Monitors → New monitor → Condition monitor**, where the escalation policy, tags, and condition are all collected up front. The condition is described inline using a sentence form for the common threshold case, or you can attach an existing condition instead. Previously, monitor creation was reached through the condition editor. Editing an existing monitor's details (policy and tags) also has its own page now. See Monitors.

- **New objective creation page** — Creating a service level objective now has its own page at **Objectives → New objective**, which collects the name, target, window, and coverage floor alongside an inline indicator description. Previously this was done inside the condition editor. See Objectives.
- **The conditions list can create a monitor for a condition** — A "Create monitor" row action on the conditions list takes an existing condition straight to the monitor create page with that condition already attached, so a second monitor on a shared condition no longer starts from the monitor list. See Monitors.

Improved

- **Trigger incident descriptions now render as Markdown** — When a trigger monitor's **Incident description** rule extracts text from the webhook payload, that text is now shown on the incident detail page as rendered Markdown. A paging system's alert body often contains links to runbooks, lists of affected assets, and other formatted context; Markdown lets that formatting come through. When the description would repeat the cause subtitle, the cause is hidden. See Incidents.

August 27, 2026

New

- **Condition editor on mobile** — Following a condition editor link on a phone now opens a read-only view of the condition: the graph lit by the appliance's live evaluation of each node, and a Details tab with the condition's description, current issues, objectives, and how many monitors use it. See Monitors.
- **Live values in the condition editor** — The condition editor's preview panel now has a "Live" tab that shows the appliance's real-time evaluation of every node in the stored condition, so you can see exactly which values are causing an alert without running a backtest. The tab is available once the condition has been saved and has no unsaved edits. See Monitors.
- **Trigger monitor source timestamp** — Trigger monitors can now be configured with a "Source timestamp" rule that extracts when the sender says the alert fired; the extracted time appears in the delivery log and is recorded in the incident's activity feed whenever it changes, so you can tell when an alert was raised at the source versus when Pulse received it. See Monitors.

Improved

- **Conditions list shows current state and usage details** — Each row on the Conditions page now shows a status indicator (Violated, Holding, Unknown, etc.), how many monitors and objectives use the condition, and how many data points it reads;

new filters let you narrow by state or to conditions not yet used by anything. See Monitors.

- **Report detail page loads older snapshots on demand** — The list of previously generated reports on a schedule's detail page now shows a "Load more" button to page through older entries, rather than loading the full history at once. See Reports.

Fixed

- **Condition editor can repair an unreadable stored graph** — If a condition's saved graph was written by a newer appliance build and this version cannot decode it, the editor now shows a clear notice and lets you draw a replacement graph and save to overwrite it, instead of being stuck. See Monitors.
- **Organization selector on mobile closes after picking** — The organization selector bottom sheet on mobile now closes automatically when you select an organization and has a dedicated Close button, so you are never left in an open sheet with no obvious way out.

August 26, 2026

New

- **Bulk selection mode on the monitor list** — Organization administrators on desktop can now click **Select** in the toolbar to enter selection mode. In this mode each row becomes a selectable target rather than a link, and a floating action bar at the bottom offers **Mute**, **Unmute**, and **Delete** for the whole selection at once. Press Escape to exit — first press clears the selection, second press leaves the mode. See Monitors.
- **Trigger monitors can now identify the sending system** — Add a "Sending system" rule to a trigger monitor so that two services sharing one webhook URL produce separate incidents even when they send the same alert ID; the sending system is named on each incident it produces. See Monitors.
- **Trigger monitors can now auto-close incidents on a reset signal** — Add a "Reset signal" rule and Pulse will immediately close all open incidents for a sender when a matching reset payload arrives. See Monitors.
- **Condition monitors now have their own sidebar entry** — Condition monitors appear directly in the sidebar so you can open the full list in one click without navigating through another section. See Monitors.
- **Search the condition library and objectives list** — Type to find a condition or objective by name instead of scrolling the full list. See Monitors.
- **Tags are now shown and filterable on the data source and CVE lists** — Each row carries the relevant tags, and the tag filter sits in each list's toolbar, so you can narrow by tag without opening the full filter menu — as the monitor and incident lists already

allow. See Vulnerabilities.

Improved

- **Incident list cards now show the triggering monitor's tags** — Each incident card in the list displays the tags from the monitor that raised the incident alongside the status badge and monitor type, so you can filter and spot patterns without opening each incident. See Incidents.
- **Monitor type chips replace the CVE quick-filter button** — A chip row below the search bar now shows every monitor type present in your list along with its count. Click any chip to filter by that type; click again to clear. See Monitors.
- **Tags filter promoted to the monitor toolbar** — The Tags filter button now sits beside the search box and type chips rather than inside the filter menu, matching its importance as the primary way to slice a large monitor list. See Monitors.
- **Clearing filters no longer clears the search term** — The **Clear filters** button in the monitor and incident list toolbars now resets only the menu-driven filters (Status, Type, Tags, Stale, Muted), leaving whatever you typed in the search box untouched. See Monitors.
- **The Filter button on every list now shows how many filters are active** — A small count badge appears on the Filter button whenever filters are applied, so a narrowed list is always visible without opening the menu.
- **Filtered empty states now explain which filters are active** — When a filter combination leaves a list with nothing to show, the empty state names the active filters and offers a one-click clear button.
- **Search terms are now kept in the URL on more lists** — Typing into a list's search box updates the URL, so a filtered view can be bookmarked or shared.
- **Detail pages adapt better to narrow screens** — On phones, the page title no longer gets squeezed by action buttons; it now stacks cleanly below the back button, and collapsed actions share the back row so neither needs its own row.
- **Destructive actions are now tucked into a `⋯` overflow menu** — Delete and other rarely-used actions on detail pages appear under a `⋯` button rather than in the main button row, so they are one click away but not one click from the work. See Monitors and Incidents.
- **Prometheus data points now carry their full series label** — When a Prometheus query returns multiple series, each data point is identified by its full label set rather than the bare metric name, so multi-series panels tell their lines apart. See Monitors.

Fixed

- **The monitor bulk action bar no longer overlaps the list rows** — The action bar that appears when monitors are selected now floats clear of the rows beneath it.

- **Dialogs no longer clip long titles or footer buttons** — Long dialog titles are kept clear of the close button; footer action buttons stay inside the dialog; and the "leave editor?" confirmation now sizes itself to fit its content.

August 25, 2026

Improved

- **Reports now include an error budget burndown chart for each objective** — Each objective section in a scheduled report now shows a chart of how its error budget was consumed day by day, so you can see at a glance whether budget loss was concentrated in a short window or spread evenly across the period. See Reports and Objectives.

Fixed

- **On the appliance, software updates that include slow database migrations now complete successfully** — Previously, an update involving a long-running migration — such as rebuilding an index on a large incident history — could fail in a restart loop: the system would time out and kill the app before the migration finished, rolling it back, then repeat until the update was aborted. The start window is now long enough for migrations to complete.

August 24, 2026

Fixed

- **Condition preview now shows data for 24 h and 7 d ranges** — The preview chart in the condition editor was rendering no data when the 24 h or 7 d time range was selected. The preview grid is now aligned to the same bucket boundaries the server uses, so all returned samples match a grid point and the chart fills in correctly. See Monitors.
- **Condition and SLO monitors now show their Activity Feed** — Their detail pages recorded every status change but showed none of them, so the history of a monitor was only reachable through its incidents. Both now show the same Activity Feed the other types do, along with their current status and the time of their last check. Nothing was lost: the entries you are seeing were being written all along. See Monitors.

August 23, 2026

Fixed

- **Navigating from list views to detail pages no longer loops back to the list** — A race condition caused the CVE and incident list headers to rewrite the browser URL just after you navigated to a detail page, leaving you on the list instead; both headers now stop syncing the URL once you have left the list.

August 22, 2026

Changed

- **Heartbeats are now data sources, and their monitors are ordinary monitors** — A heartbeat is a push data source: Pulse acquires one signal from it, "a ping arrived", and a condition monitor alerts when that signal goes quiet for longer than the interval you set. Your existing heartbeats were converted for you and keep their name, their tags, their escalation policy, their incident history and their link. **The ping URL has not changed** — nothing you have deployed needs reconfiguring. Creating one is still the same two fields, now under Data sources; the ping URL and the rotate button live on the heartbeat's page there. See Heartbeat Monitors.
- **A heartbeat can now be combined with anything else** — Because the ping is an ordinary signal, "the nightly backup has not checked in **and** it is a weekday" is one condition rather than something you could not express. Open the monitor a heartbeat created and edit its condition like any other. See Monitors.
- **Heartbeats still cost nothing** — A heartbeat's data source and its one data point are excluded from your licence's provisioned counts, exactly as heartbeat monitors have always been. Your numbers are unchanged.
- **The sidebar is now Data sources, Monitors, Objectives** — Three sections for the three things Pulse does: data sources acquire, monitors alert, objectives promise. **Data sources** carries the per-protocol entries — OPC UA, S7 and Prometheus — so the entry you used every morning to find your PLCs is still one click away, just one section over, and now shows the connection and everything read through it rather than a list of thresholds. **Monitors** opens one flat list, with shortcuts kept only for CVE, Trigger and SLO. Old links to the OPC UA and S7 monitor lists take you to the matching data sources. See Monitors.
- **Tags filter the monitor list from a button of their own** — With the per-type sidebar entries gone, tags are the main way to narrow the list, so the tag picker sits in the toolbar beside the search box instead of inside the filter menu. A condition monitor inherits the tags of the data it reads, so tagging a PLC once is still enough. See Monitors.
- **Conditions moved to the bottom of the sidebar** — The conditions library is beside Tags and Severities now. You still open a condition by opening the monitor that alerts on it; the library is for the case where several monitors, or an objective, share one. See

Monitors.

- **OPC UA and S7 connections are now added as data sources** — A PLC or an OPC UA server is added under Data sources, with its address, credentials, certificate and security settings in one place — the same way a Prometheus endpoint already was. There is no separate "endpoint monitor" to create first. Existing connections were moved for you and are already listed there. See OPC UA and S7.
- **A new data source alerts you when it goes unreachable, unless you say otherwise** — Adding a source creates a condition monitor called "<source> unreachable" alongside it. It is an ordinary monitor: rename it, mute it, change its escalation policy, edit its condition or delete it. Untick the box while adding the source if you do not want one. See OPC UA and S7.
- **Browsing a server's nodes and pinning its certificate moved with the connection** — Both now live on the data source rather than on a monitor, which is where the connection they use has been since the sources were introduced.
- **CVE monitors discovered from a device now name that device's data source** — The link is the same one; what it points at is the source rather than the monitor that used to stand for it.
- **OPC UA and S7 endpoint monitors are now condition monitors** — Every existing endpoint monitor has been converted in place. It keeps its name, its tags, its escalation policy, its incident history and its link, and it now says what it always meant: "alert when this data source cannot be reached". It opens in a simple form, not on a diagram. Nothing about the upgrade asks you to do anything. See Monitors.
- **You can now combine a value with its source's reachability** — A condition can read a source's reachability alongside its readings, so "alert when the boiler is above 80 °C **and** the PLC is answering" is one condition. An unplugged cable then raises the one incident that says the cable is unplugged, instead of one per value being read through it. See Monitors.
- **Converted endpoint monitors are checked every five seconds** — An endpoint monitor had its own check interval; a condition does not, because conditions are evaluated on Pulse's five-second schedule. Detection is faster, and a connection that drops in and out will open and close incidents more often than it used to. If that is noisier than you want, open the condition and add a "sustained for" step so it only alerts once the source has been unreachable for a while. See Monitors.
- **A converted monitor stays quiet while Pulse itself cannot answer** — An endpoint monitor used to raise an incident when Pulse's own OPC UA or S7 service failed, which read as the device being down when it was not. A converted monitor reports what it knows: it alerts when the device refuses or times out, and stays as it is when Pulse has no recent verdict at all.

- **A custom incident message on an endpoint monitor is now the condition's alert description** — If your message was plain text, it has been carried across word for word. If it contained a placeholder — an address wrapped in doubled curly braces — it has not: an alert description is shown exactly as written, so the placeholder would have reached you unfilled. Incidents raised before the upgrade are unaffected. See Monitors.

Fixed

- **On-call handoff emails now show the correct message** — Plain-text on-call handoff emails were sending both the shift-starting and shift-ending body to every recipient due to a template rendering bug; recipients now receive only the message that matches their actual handoff direction.
- **Report-ready emails no longer include mismatched access instructions** — Plain-text report-ready emails were including both the password-protected and public access sections regardless of whether the report had a password; each email now contains only the section that applies.
- **Deleting monitors or objectives no longer leaves sensors permanently stuck** — When a monitor or objective backed by a condition was deleted, the condition record was sometimes left behind; that orphaned condition would then block deleting any sensor it had read. Conditions whose last consumer is deleted are now cleaned up in the same operation.
- **In the mobile app, opening or refreshing the app no longer shows a server error** — A conflict between device token registrations could cause a unique-constraint error on every app launch, surfacing as a persistent "server error" toast; the registration now clears any stale claim on the token before writing it.

Removed

- **Endpoint, node and tag monitors can no longer be created** — All three were converted into condition monitors over the last two releases, and the forms and API endpoints that created them have been removed along with the tables behind them. Conditions do everything they did and combine freely; a value and its source's reachability in one rule is the case that was not expressible before. See Monitors.
- **Deleting a monitor no longer cascades, and policies and tags no longer apply "to children"** — A monitor has no children any more, so the confirmation step that offered to delete them, and the prompts that offered to copy an escalation policy or a set of tags down to them, are gone. Tags on a data source still flow to everything read through it.
- **Importing monitors from CSV now covers CVE monitors only** — The OPC UA and S7 rows created monitor types that no longer exist. Build those as data sources and conditions instead.

August 21, 2026

New

- **Bulk mute and delete monitors** — The monitor list now lets you select multiple monitors at once and mute, unmute, or delete them all in a single action; a selection bar appears when monitors are chosen, with accelerators to select the full list. See Monitors.
- **Incidents reopen automatically within the escalation policy window** — Escalation policies now have a reopen window setting; if a resolved incident's trigger fires again before the window expires, Pulse reopens the existing incident instead of creating a new one, and holds the resolved notification until the window passes. See Escalation Policies and Incidents.

Improved

- **Deleting an endpoint monitor now removes all its sub-monitors in one step** — Previously, deleting an OPC UA or S7 endpoint monitor was blocked while node or tag monitors were attached to it; the delete dialog now shows the full count of monitors that will be removed and deletes everything together after you confirm by typing the monitor's name. See Monitors.

Changed

- **OPC UA node monitors and S7 tag monitors are now condition monitors** — Every existing node and tag monitor has been converted in place. It keeps its name, its threshold, its tags, its escalation policy, its incident history and its link, and it opens in the same simple form as before — "alert when the average of this value over the last five minutes is above 80" — with the option, new to it, of adding a second reading or combining it with the source's reachability. Nothing about the upgrade asks you to do anything. Creating a new one starts in the condition editor instead of the old node/tag form. See Monitors.
- **Converted monitors are checked every five seconds** — A node or tag monitor had its own check interval; a condition does not, because conditions are evaluated on Pulse's five-second schedule. A monitor that used to be checked every five minutes is now checked every five seconds. Detection is faster, and a value that crosses its threshold repeatedly will open and close incidents more often than it used to. If that is noisier than you want for a particular monitor, open its condition and add a "sustained for" step so it only alerts once the reading has held. See Monitors.
- **Reports filtered by monitor type now cover all of your condition monitors** — A scheduled report scoped to "OPC UA node" or "S7 tag" was filtering on a monitor type that no longer exists, so its filter has been rewritten to "condition". Without that it would have kept sending on schedule and arrived empty. Converted monitors and

conditions you drew yourself are the same type now, so that filter no longer tells them apart, and a report scoped this way will include condition monitors it did not cover before. To narrow it again, open the report and add a tag filter — your converted monitors kept their tags. See Reports.

Fixed

- **"Sum" and "count" aggregations now count your readings, not time slots** — A condition using the sum or count aggregation was folding Pulse's internal evaluation grid rather than the readings themselves, so a reading was counted again for every slot it was still the most recent one. The practical effect: a sensor that stopped reporting still produced a full count, so a "count is below N" condition — the usual way to alert on a sensor going quiet — stayed silent exactly when it should have fired. Sum and count now add each reading once. Expect these two aggregations to report **lower** numbers than before; if you tuned a threshold against the old value, review it. Minimum, maximum and average are unaffected. Two steps are deliberately not affected by this: a "Fallback" still contributes every value it substitutes for missing data, and a "Has data" still reports one answer per time slot — so gap-filling and uptime conditions keep counting slots, which is what they are for. See Monitors.
- **Escalation policy deletion now correctly checks whether the policy is in use** — A bug that allowed deletion of escalation policies still referenced by monitors or incidents — while blocking deletion of unused ones — has been fixed. See Escalation Policies.

August 19, 2026

Improved

- **Condition nodes show which value an incident will quote** — Auto-generated conditions (such as band detectors created by the anomaly-detection action on sensor monitors) now label each relevant node — Observed, Upper limit, or Lower limit — on the node card and in the condition preview on the monitor page, so you can see at a glance which number an incident is about without opening the editor. Selecting a tagged node in the inspector shows the same label under "Quoted in incidents as". See Monitors.
- **Condition editor validates nested time-window chains** — When aggregate nodes are wired in series (for example, an average feeding another average), the editor now checks whether the combined lookback exceeds Pulse's data retention period and highlights the first node where the total crosses the limit, so you know exactly which one to shorten. See Monitors.

Fixed

- **Anomaly detection baseline runs now always complete** — A bug where the final data chunk of every baseline run failed to enqueue the finalization step has been fixed; baselines now finish reliably and produce their full result. See Anomaly Detection.

August 18, 2026

New

- **Service Level Objectives** — The new Objectives section lets you define a promise over a condition — for example, "this must hold at least 99.5 % of the time over the past 30 days" — track its error budget, and see at a glance whether the objective is being met. Add burn alerts to get paged when the budget burns too fast. See Objectives.
- **Multiple named alerts in the condition editor** — A single condition graph can now contain several Alert nodes, each with its own incident name and description, so one condition can raise distinctly-named incidents for different failure modes.

Improved

- **Sidebar shows only your teams** — The Teams section in the sidebar now lists only the teams you belong to, and hides itself entirely when you are not a member of any team.

Fixed

- **The app no longer logs you out on unrelated 401 responses** — Session expiry is now detected precisely: the app ends your session only when the authentication check confirms the token is dead, not when a resource-level request is refused with a 401.

August 17, 2026

New

- **Detect anomalies on sensor monitors** — OPC UA and S7 sensor monitors now have a "Detect anomalies" action that creates condition monitors watching for the sensor dropping out, reading outside its learned normal range, holding flat, or varying unusually; each check starts in shadow mode (recording what it would have done without alerting) so you can review a week of results before activating. See Anomaly Detection.

Improved

- **AI Canvas board and chat are now independent panes** — The board holds only the assistant's findings as persistent cards you can drag, resize, and arrange freely; the chat panel alongside it shows the complete conversation including tool calls; and an empty board now shows a note explaining what will appear there once the assistant finds something. See AI Canvas.
- **Filter monitors and incidents by Condition monitor type** — The type filter on the monitor list and incident list now includes "Condition monitors" (and "Trigger monitors" on the incident list), so you can focus on those monitors directly. See Monitors and Incidents.
- **Incident escalation pauses while a monitor's verdict is unknown** — When a monitor cannot be evaluated — for example because its sensor has stopped reporting data — escalation now holds rather than continuing to page on-call responders; it resumes automatically once the monitor delivers a definite verdict. See Incidents.

Fixed

- **Query page charts no longer flicker when moving the crosshair** — Hovering the crosshair no longer triggers a full repaint of all chart series, eliminating a visible stutter when moving the pointer across data-dense charts. See Query.

August 15, 2026

Improved

- **Condition editor panels can be collapsed** — The palette and inspector side panels in the condition editor now have collapse buttons that fold each to a narrow strip, so the graph gets more room when you need it. See Conditions.
- **Condition editor remembers your layout** — The fold state of both side panels, the graphs toggle in the preview, and the preview panel height are now saved in your browser, so the editor opens exactly the way you left it. See Conditions.
- **Sensor charts show what your condition actually compares** — When a sensor feeds an Aggregate node, its chart in the condition preview now draws the aggregate's output alongside the raw readings — the value your Compare node tests — computed with the same bucketing the condition evaluation uses. See Conditions.
- **Sensor charts hold their place while new data loads** — The preview panel no longer collapses and reflows when you change the time range; each chart shows a placeholder while data is in flight.
- **Hovering a sensor chart highlights it on the canvas** — Moving the pointer over a sensor chart in the condition preview now highlights the corresponding nodes on the canvas.

- **Query page crosshair spans all aggregation charts** — The vertical reference line on the Query page now stays visible as the pointer moves between aggregation charts, including across the headings and gaps between them, making it easy to compare values from different aggregations at the same instant. See Query.

August 14, 2026

New

- **Conditions list page** — A new Conditions page lists all your saved conditions in one place, shows how many monitors use each one, and lets you create new conditions or delete unused ones.
- **Condition monitors raise and resolve incidents** — Pulse now evaluates condition monitors on a regular schedule; when a condition starts alerting it opens an incident through your escalation policy, and when it clears the incident is resolved automatically.

Improved

- **Condition Monitors in the sidebar** — The monitor navigation now includes a Condition Monitors entry so you can jump straight to monitors backed by your conditions, alongside OPC UA, S7, and other monitor types. See Monitors.

Fixed

- **Deleting a monitor used by a condition now shows a clear error** — Trying to delete a monitor that a condition depends on now explains the block and asks you to remove the monitor from its condition first, instead of failing silently.

August 13, 2026

New

- **"Latest" aggregation in PQL** — The query builder now offers a "Latest" aggregation that returns the most recent sensor value in each time bucket, letting you see the last-recorded reading per interval rather than a statistical summary. See Query.

August 12, 2026

New

- **Heartbeat monitors** — A new monitor type for watching periodic jobs: configure an expected ping interval, point your cron job or scheduled task at the generated URL, and Pulse opens an incident when the ping stops arriving on schedule. The monitor's

detail page shows the last ping time, a copyable ping URL, and a one-click URL rotation to invalidate old credentials. See Heartbeat.

- **Condition editor** — A new visual canvas for authoring monitor alert conditions: place and wire together typed nodes — sensors, monitor state, arithmetic, comparisons, and logic gates — then preview how the condition would have fired against historical data, with a text mirror strip rendering the full expression.

August 11, 2026

New

- **Earlier occurrences shown on the incident detail page** — The incident detail page now lists up to 10 earlier incidents caused by the same monitoring condition, each showing when it happened and how long it lasted, so you can tell at a glance whether the problem is recurring. See Incidents.

Improved

- **Warning before resolving an incident while the monitor is still unhealthy** — When you resolve an incident while its monitor is still failing, Pulse now shows a warning that a new incident will be raised within seconds; click "Resolve anyway" to proceed. See Incidents.
- **Monitor and CVE list filters on mobile use a full-screen drawer** — The monitors and vulnerabilities list pages now use the shared filter drawer on mobile, consistent with the incident list: tap Filter to open all filter groups, stage your selections, then tap Apply. See Monitors and Vulnerabilities.
- **Query chart tooltips show the full date and time** — Data point tooltips in the Query chart now display the day, month, and year alongside the time, so you can tell which date a point falls on when the query spans multiple days. See Query.
- **Report recurring-monitors list capped at ten** — The "Top recurring monitors" section in a report snapshot now shows at most ten monitors and notes how many were omitted, keeping the panel a consistent size. See Reports.

Fixed

- **Cleared incident status filter no longer resets to the default** — Removing all status filters from the incident list and navigating away no longer snaps the selection back to Unacknowledged + Acknowledged; the explicitly cleared state is now preserved in the URL. See Incidents.

August 10, 2026

Security

- **Report snapshots are now encrypted end-to-end** — When Pulse sends a report link by email, the report content is sealed in an encrypted envelope before it reaches the gateway; only the intended recipient can unlock it with the password from their email, and no password hashes are stored or transmitted anywhere. See Reports.

August 9, 2026

New

- **Open-incident count badges in the sidebar** — The Incidents and My Incidents navigation entries now show a live badge with the number of currently open incidents, so you can see at a glance how many incidents need attention without navigating to the incident list. See Incidents.

Improved

- **Filter PQL queries by tag** — The query builder's data point selector now lets you pick tags alongside individual monitor endpoints; the query returns aggregated data for all sensors carrying those tags, making it easy to scope a chart to a machine group or site without selecting endpoints one by one. See Query.

August 8, 2026

Improved

- **PQL queries seamlessly span historical and live sensor data** — Sensor data is now exported to a local analytics lakehouse every ~20 seconds, and PQL queries read from it in a single sweep that covers both the archived history and the live tail, eliminating any boundary between historical and recent results. See Query.
- **On the appliance, backups now include analytics lake data** — The appliance backup command now captures the analytics lake directory alongside the database dump, and restores it on recovery, so historical sensor time-series data is preserved across backup cycles.

Fixed

- **Incident markers on the Query page appear with the first result** — Incident overlay markers are now fetched before the query executes, so they appear in the same response as the results rather than blinking in after a second render. Re-running an

unchanged query now always dispatches a fresh request. See Query.

August 6, 2026

Improved

- **On the appliance, re-applying the current bundle now skips the backup and import** — When the update bundle already matches the running system closure, the appliance now detects this immediately after bundle verification and completes in seconds, skipping the database backup and the multi-gigabyte system image import that previously ran before reaching the same "Already up to date" result.

Fixed

- **On the appliance, System Status now shows the real uptime** — The System Status screen now displays the actual uptime (for example, "2h 4m" or "1d 2h 4m") instead of always showing "unknown". The appliance's `uptime` binary does not support the flag that was previously used, so the screen now reads `/proc/uptime` directly.
- **On the appliance, System Status fallbacks are now localized** — The fallback text shown when uptime or disk usage cannot be read now appears in the operator's language (for example, "unbekannt" in German) rather than always in English.
- **On the appliance, update failure messages are now fully in the operator's language** — The update failure screen now shows error messages end-to-end in the operator's language. Previously some messages had a translated outer label wrapping an untranslated inner detail (for example, "Überprüfung fehlgeschlagen: checksum mismatch: ...").
- **On the appliance, signature verification errors no longer repeat "signify:"** — When bundle signature verification fails, the error message now shows the "signify:" label exactly once instead of doubling it (previously it read "signify: signify: ...").
- **On the appliance, the staged-update screen no longer instructs to reboot twice** — After staging an update that requires a reboot, the reboot instruction now appears exactly once (in the done screen footer) instead of repeating in the progress history.

August 5, 2026

Improved

- **Suggested queries open as a bottom drawer on mobile** — On small screens, the suggested-queries panel on the Query page is now a bottom drawer that shows each query's name and description inline, making it easier to browse and run queries by

touch. See Query.

- **CVE severity chart labels readable on mobile** — The severity bar chart on the CVE dashboard now rotates its axis labels on small screens so they no longer overlap. See Vulnerabilities.
- **On the appliance, installer and update progress is now in German** — Progress messages during installation and software updates now appear in German on the operator's screen, while the audit log and install log remain in English for support engineers.
- **On the appliance, re-applying the running version reports "Already up to date"** — When an update bundle contains the closure already running on the appliance, the update flow now correctly reports that no change was made, instead of showing a misleading completion message.

Fixed

- **Incident list rows navigate reliably** — The incident list rows have been rebuilt with a correct click-target structure so the full row navigates to the incident detail; the Actions dropdown and status badge no longer accidentally trigger row-level navigation. See Incidents.
- **Escalation step cards show the step number** — Each step card in the escalation policy editor now displays its number ("Step 1", "Step 2", etc.) instead of the generic label "Step". See Escalation Policies.
- **Input fields with button addons no longer overhang at desktop widths** — A layout bug where an addon button created an invisible hit-testable strip outside the field boundary on desktop screens has been fixed.
- **Email action links now reach Pulse directly** — Action links in incident alert, digest, on-call handoff, and report emails now carry `clicktracking="off"` so email providers do not rewrite them through a tracking redirect — one-click actions such as Acknowledge continue to work regardless of which email service delivers the message.

August 4, 2026

New

- **Mobile incident actions** — On small screens, the incident detail page now shows a large Acknowledge or Resolve button pinned to the bottom of the screen with a confirmation drawer to prevent accidental taps, so the primary action is thumb-reachable without opening the Actions dropdown. See Incidents.

Improved

- **Compact incidents toolbar on mobile** — On small screens, the search field is now hidden behind a toggle icon that focuses the input when tapped, the Create Incident button moves to a floating button in the bottom corner, and the filter button shrinks to an icon, keeping the toolbar uncluttered. See Incidents.
- **Query builder suggestion lists work correctly on small screens** — The suggestion dropdowns in the Query Builder no longer get clipped at the top of the screen when space is tight, and each option has a larger touch target on mobile. See Query.

Fixed

- **Scheduled reports include only incidents from the selected period** — A bug where incidents outside the report's time window appeared in scheduled incident reports has been fixed. See Reports.

August 2, 2026

Improved

- **Incident status shown as a labeled badge** — The colored dot on incident list cards is replaced by a text badge that names the status (for example, "Unacknowledged"), so the status is readable at a glance without relying on color alone. See Incidents.
- **App header stays fixed on mobile while scrolling** — The top navigation bar is now sticky on mobile screens, keeping the menu trigger and controls accessible no matter how far you scroll.
- **Report actions consolidated into a dropdown** — The "Generate now" and "Delete" actions on a report's detail page are now grouped under a single "Actions" button, reducing clutter in the header. See Reports.

Fixed

- **Redundant incident cause no longer shown** — When a monitor's cause text matches the incident title exactly, it is no longer repeated on the incident list card or detail page. See Incidents.
- **Drawers no longer scroll-jump the page on iOS** — In the mobile app, opening a drawer (such as the comment panel) no longer caused the page behind it to jump; buttons inside drawers also respond correctly again, and tapping the overlay beside a drawer reliably closes it.

August 1, 2026

New

- **Propagate tag changes to child monitors** — When you save changed tags on an OPC UA endpoint or S7 endpoint monitor, a dialog now offers to apply the same additions and removals to all child monitors at once, so a parent's tags and its children's tags can be kept in sync without editing each one separately. See Monitors.

Improved

- **Trigger monitor shows a unified activity feed with delivery details** — The trigger monitor detail page now merges webhook deliveries and status-change events into a single chronological feed; burst runs of deliveries collapse into a grouped entry you can expand, and clicking "Details" on any delivery opens a panel with the full payload (with a formatted JSON toggle and copy button), content type, linked incident, and prev/next navigation. See Monitors.
- **Reports can now be sent to teams** — The report form's recipient section is renamed "Pulse users and teams" and now lets you add entire teams as recipients; team members are resolved each time the report is generated. See Reports.
- **Report deletion now requires confirmation** — Deleting a report now shows a confirmation dialog explaining that no further reports will be generated and that already-sent reports remain accessible via their existing links, preventing accidental deletion. See Reports.
- **Large report incident tables scroll smoothly** — The incident table in a report snapshot is now virtualized, so reports with hundreds or thousands of incidents render and scroll without slowdown. See Reports.

August 3, 2026

Improved

- **Incident filters on mobile use a full-screen drawer** — On mobile, tapping the filter button opens a drawer with all filter groups (status, type, monitor type, tags, acknowledged by, resolved by) as collapsible sections; selections are staged and committed only when you tap Apply, and active filters appear as removable chips below the toolbar. See Incidents.
- **On-call calendar has a dedicated view switcher on mobile** — A labeled dropdown now lets you pick Day, Week, Month, or Year view on small screens, and the calendar remembers your mobile and desktop view preferences separately. See On-call Schedules.
- **Buttons, inputs, and menus have larger touch targets on mobile** — All interactive controls now meet the 44 px minimum tap-area requirement on touch screens, and text fields use at least 16 px font size to prevent iOS Safari from zooming when a field is focused.

Fixed

- **Back navigation works when pages are opened from external links** — The back button on monitor, monitor-edit, and incident detail pages now navigates to the relevant list page when you arrived via a direct link (such as from a notification or bookmark), instead of leaving the app.

July 31, 2026

New

- **Propagate escalation policy to child monitors** — After saving an endpoint monitor with a changed escalation policy, a dialog now offers to apply that same policy to all child monitors at once, so parent and children stay in sync without editing each one individually. See Monitors.

Improved

- **Tag list groups tags by category** — The tags page now shows tags organized into collapsible category sections with filter chips at the top, so you can focus on one category at a time; searching still shows a flat list across all categories. See Tags.
- **Trigger monitor shows when the last webhook arrived** — The ingest URL card on a trigger monitor's detail page now displays how long ago the most recent delivery was received, so you can confirm the integration is active without opening the full deliveries log. See Monitors.
- **Monitor event history shows 20 events per page** — The event history on the monitor detail page now loads 20 events at once instead of 5, so you can see a fuller picture of recent activity without paginating. See Monitors.

Fixed

- **Push notification taps and tray actions now work reliably in the mobile app** — Several fixes to how notifications are handled ensure that tapping a push notification opens the correct incident, tray actions (Acknowledge/Resolve) reliably reach the gateway, failed actions show an error alert rather than silently disappearing, and offline actions are automatically retried when connectivity returns.

July 30, 2026

New

- **Scheduled incident reports** — Admins can now create recurring incident report schedules (daily, weekly, or monthly), filter included incidents by status, monitor type, or tag, add external email recipients and Pulse users, and configure how long shared links stay valid. Reports are emailed automatically on schedule; recipients open them via a password-protected public link. See Reports.
- **Acknowledge or resolve incidents from the iOS notification tray** — In the mobile app, push notifications for open incidents now show Acknowledge and Resolve action buttons; tapping one forwards the command to Pulse even when the app is offline, so the incident is acted on without opening the appliance.

Improved

- **Push notifications can navigate to more pages** — In the mobile app, tapping a digest or handoff notification now opens the correct destination (e.g. the incident list or the on-call calendar) instead of only navigating to the incident detail page.

Fixed

- **Pull-to-refresh no longer redirects to the dashboard** — In the mobile app, pulling to refresh a page no longer dropped the user back to the dashboard; the app now stays on the current page by reinjecting the auth token without remounting.

July 29, 2026

New

- **Incident Reports — scheduled summaries delivered to your team** — The new Reports page (`/app/reports`) lets admins create scheduled incident reports that run daily, weekly, or monthly. Each generated report is frozen at the moment it runs and covers headline counts, response times (MTTA/MTTR), volume trends, onset heatmap, monitor and tag breakdowns, responder load, and a filterable incident table. Pulse users receive an in-app notification and email linking to an authenticated in-app view; external recipients receive a personal shared link protected by a one-time password. See Reports.
- **Tag categories — group related tags into dimensions** — Organization admins can now create tag categories (e.g. Customer, Project, Site) that group related tags together; each category can be configured as single-select (a monitor may carry only one tag from that category) or multi-select. Tags can be assigned to a category when created or edited, and the tag selector on a monitor now groups options by category and shows co-occurrence counts to help you pick related tags quickly. See Tags and Tag Categories.

- **Forgot-password / reset-password flow** — The sign-in page now has a "Forgot password?" link that lets users request a password-reset email; following the link in the email opens a reset form where they can choose a new password. The reset link expires after one hour.
- **Filter incidents by who acknowledged or resolved them** — The incident list filter menu now includes "Acknowledged by" and "Resolved by" dropdowns so you can scope the list to incidents a specific team member handled. A new "My Incidents" entry in the sidebar links directly to the incidents you acknowledged. See Incidents.

Improved

- **Incident list shows who acted on each row** — Acknowledged and resolved incidents now display a small initials badge next to the row; hovering shows the person's name and when they acted, and clicking the badge re-scopes the list to that user's incidents. The incident detail page similarly shows an initials avatar for the acknowledger or resolver, linked to their filtered list. See Incidents.
- **Tag filter groups options by category** — When filtering monitors, incidents, or CVE findings by tag, the tag submenu now groups options under their category heading and shows a co-occurrence count next to tags that match the current selection, so the most relevant choices sort to the top.
- **Tag badges show their category in the tooltip** — Hovering a tag badge on a monitor or incident now shows which category the tag belongs to (e.g. "Category: Customer") alongside any description. See Tags.
- **AI Canvas can list tags** — The AI assistant can now return a list of tags as a canvas card, with each row linking to the monitor list filtered by that tag. See AI Canvas.
- **CVE provider status distinguishes polling from downloading** — The CVE status page now shows a "Last polled" timestamp alongside "Last attempt", so administrators can tell whether a provider's feed is being checked regularly (even when no advisories changed) versus genuinely unreachable. See Vulnerabilities.

July 28, 2026

Improved

- **Verification email has a refreshed design** — The account verification email now uses the same visual style as other Pulse system emails, with a cleaner layout and consistent branding.

July 27, 2026

Improved

- **Updated Pulse branding across web and mobile apps** — The web app now shows the Pulse wordmark in the sidebar and on sign-in pages alongside an updated favicon; in the mobile app, the app is now named "Pulse Dispatch" with updated icons including light, dark, and tinted variants on iOS.
- **AI Canvas step cards no longer show connector arrows** — The curved arrows that previously linked step cards in execution order have been removed, leaving a cleaner canvas layout with only the numbered sequence badges. See AI Canvas.

Fixed

- **S7 monitor connections are more reliable** — Connection attempts to S7 devices are now handled per endpoint with deduplication, preventing redundant simultaneous dials and eliminating potential stalls when multiple monitors target the same PLC. See S7.

July 25, 2026

New

- **Trigger monitors — raise incidents from inbound webhooks** — A new Trigger monitor type generates a unique ingest URL that external systems (Grafana, Alertmanager, custom scripts) can POST to; Pulse parses the payload using configurable rules (regex, JSONPath, or text matching) to extract an alert ID, incident title, and description, and can automatically acknowledge or resolve the incident when a matching payload arrives. Optional request-header matchers let you put your own shared-secret authentication in front of the ingest URL. The monitor detail page shows the ingest URL with copy and rotate actions, plus a recent-deliveries log showing how each incoming webhook was handled. See Monitors.

July 21, 2026

New

- **Filter incidents and CVE findings by monitor tag** — The Incidents and Vulnerabilities list filters now include a Tags option, so you can narrow results to the monitors carrying a specific label without having to know which monitors that covers. See Incidents and Vulnerabilities.

July 19, 2026

Improved

- **Vulnerabilities page reorganized into a single flat tab bar** — The CVE page now has four top-level tabs — Overview, CVEs, Findings, and Activity — replacing the previous layout where the dashboard and a grouped/flat view toggle were nested inside a single Findings tab. See Vulnerabilities.

July 18, 2026

New

- **Canvas result cards link directly to monitor, incident, and CVE detail pages** — Monitors, incidents, and CVE findings returned by the AI assistant are now clickable rows that open the relevant detail page, so the canvas ends where your next action starts. See AI Canvas.
- **Canvas run status shown as a colored badge in the list and switcher** — A red dot marks a failed canvas and a pulsing amber dot marks one that is still running, replacing the plain gray text label that was easy to miss. See AI Canvas.

Improved

- **AI assistant presents times in the organization's timezone** — When the AI reports times in prose, tables, or reports it now converts them to your organization's local timezone (including the correct daylight-saving offset for each date), so German operators see Central European Time instead of raw UTC. See AI Canvas.
- **Canvas titles are generated in the language of the conversation** — The auto-generated canvas title now matches the language you used in the conversation; a canvas started in German receives a German title instead of an English one. See AI Canvas.
- **Canvas tool cards show human-friendly names and icons** — The AI tool result cards on the canvas now display "Monitors", "Incidents", "Data query", or "CVE search" with a matching icon instead of the raw internal tool name. See AI Canvas.
- **Long result lists in canvas cards are now capped with a "Show more" toggle** — Lists of monitors, incidents, or CVE findings preview the first five rows; a button reveals the rest without growing the card to an unreadable size. See AI Canvas.
- **Viewing another user's canvas now shows a read-only notice** — When you open a canvas that belongs to a colleague, the chat pane explains it is read-only, names the owner, and offers a one-click link to start your own canvas. See AI Canvas.

- **Typed message is preserved when the assistant is still working** — If you try to send a follow-up while a canvas run is in flight, your message stays in the input and a notification explains why it was not sent, so nothing is silently lost. See AI Canvas.
- **Expanded CVE advisory coverage** — The CSAF aggregator now includes CERT@VDE and 39 additional vendor feeds, increasing the number of industrial CVEs surfaced in the Vulnerabilities page. See Vulnerabilities.

July 15, 2026

New

- **Vulnerabilities list now groups findings by CVE** — The Findings tab defaults to a "CVEs" view that groups all findings of each vulnerability across your devices into a single expandable row, showing severity, affected device count, and a triage status breakdown; click a row to expand its individual findings inline. A toggle switches to the flat per-finding "Findings" view. See Vulnerabilities.
- **Bulk triage all findings of a CVE** — Organization admins can now apply one triage decision to every finding of a given CVE across all devices at once using the "Triage all" action in the CVE row's menu; the dialog shows how many findings will be updated and warns when active filters make the visible count a subset of the full org-wide set. See Vulnerabilities.

July 14, 2026

Improved

- **CVE finding detail page now shows all affected devices** — The vulnerability detail page now includes an "Affected devices" panel listing every monitor in your organization with a finding for the same CVE, each with its triage status, so you can assess the full blast radius from a single page. See Vulnerabilities.

July 13, 2026

New

- **Switch between canvases without leaving the editor** — A new switcher panel in the AI Canvas editor (layers icon in the header) lists all your canvases so you can jump between them without returning to the canvas list page. See AI Canvas.

- **AI Canvas chat and canvas panes are now accessible on mobile** — On small screens, a Canvas / Chat tab bar lets you switch between the spatial canvas and the conversation pane. See AI Canvas.

Improved

- **AI Canvas list shows loading placeholders and handles errors gracefully** — The canvas list now renders placeholder rows while data loads instead of flashing an empty state, shows a retry option when loading fails, and reveals a filter input once your list grows long. See AI Canvas.
- **Canvases are named from the first prompt immediately** — When you send your first message, the canvas title is seeded from that prompt so it is identifiable while the run is in flight or if it fails; a successful run still upgrades the name to a model-generated title. See AI Canvas.
- **CVE advisory data is more reliably fetched** — The CSAF fetcher now tracks each advisory document individually, retrying transient download failures on every run so gaps from past network errors are backfilled rather than silently missed.

July 12, 2026

Improved

- **OPC UA monitor connection settings moved to a dedicated side panel** — Endpoint URL, authentication type, security configuration, and certificate pinning are now configured together in a "Configure connection" side panel that opens from a compact read-only summary card on the edit form, preventing partially-committed connection settings when switching between configurations. See OPC UA.
- **Unpinning an OPC UA server certificate now requires confirmation** — Removing a pinned certificate from the monitor edit form or the monitor detail page now shows a confirmation dialog that explains the security implications, preventing accidental removal. See OPC UA.
- **OPC UA connection failures now show specific, actionable incident descriptions** — When an OPC UA monitor cannot connect, Pulse now raises an incident with a specific cause — server unreachable, security handshake failed, certificate rejected, authentication failed, or timeout — instead of a generic error. See OPC UA.
- **OPC UA certificate rotation panel labels the pending thumbprint** — In Organization Settings, the pending replacement certificate's thumbprint is now labeled to distinguish it from the active one, and discarding a pending replacement requires a confirmation step. See OPC UA.

July 11, 2026

New

- **Pin an OPC UA server certificate by file upload or PEM paste** — The certificate pinning section on OPC UA endpoint monitors now lets you upload a certificate file (PEM or DER) or paste a PEM directly, so you can pre-configure the pin even when the OPC UA server is not reachable at setup time. See OPC UA.

Improved

- **OPC UA monitor warns when credentials are exposed without a pinned certificate** — When you use username/password authentication on a signed or encrypted channel without a pinned certificate, the monitor edit form now shows a warning that a man-in-the-middle attack could intercept the credentials. See OPC UA.
- **OPC UA monitor warns before removing a stale pin** — When you change the endpoint URL on a monitor that already has a pinned certificate, the form now shows an alert explaining that the pin will be removed on save, so the change is never silent. See OPC UA.
- **Segmented controls have a new track-and-pill appearance** — Joined toggle groups (used, for example, on OPC UA security mode and policy selection) now render as individually-rounded pills inside a muted track, making the selected option more visually distinct.

Fixed

- **OPC UA certificate error messages no longer appear twice** — When a pin or upload operation fails with an error the app already handles centrally, the error notification now appears once instead of twice. See OPC UA.

July 10, 2026

New

- **OPC UA application instance certificate management** — A new "OPC UA Certificate" tab in Organization Settings shows the certificate Pulse presents to OPC UA servers on secure channels, with its SHA-1 thumbprint for cross-checking, a PEM download, and actions to rotate the certificate (mint a replacement for pre-trust while the current one stays active), complete a rotation by cutting over, or revoke the active certificate in an emergency; OPC UA monitor detail pages also show a compact link to this tab when the monitor uses a secure channel. See OPC UA.

- **OPC UA certificate expiry alerts** — Pulse now raises an incident and emails organization admins 60 days before the shared OPC UA application instance certificate is due to expire; a pending replacement is minted automatically so its thumbprint is available for pre-trust, and Pulse promotes it automatically as a backstop once the certificate actually expires. See OPC UA.

Improved

- **On the appliance, updates are more resilient** — The post-update health check now waits up to five minutes (previously 60 seconds), reducing unnecessary rollbacks when database migrations take time to run; updates where only peripheral services fail but Pulse itself is healthy no longer trigger a rollback; and when a generation switch requires a reboot to take effect, the TUI now tells you to reboot rather than appearing to fail.
- **CVE report emails now have an HTML layout** — The weekly vulnerability report email now includes a styled HTML version alongside plain text, with color-coded counts for open, in-progress, resolved, and risk-accepted findings.

July 9, 2026

New

- **OPC UA endpoint discovery** — When creating or editing an OPC UA monitor, you can now click "Discover endpoints" to query the server and see the security configurations it actually supports; Pulse pre-selects the strongest option and shows the server certificate with subject, issuer, validity window, and thumbprint (with an expiry warning when it is near or past its expiry date). See OPC UA.
- **Server certificate pinning for OPC UA monitors** — Pulse can now pin the discovered server certificate to an OPC UA monitor: on signed or encrypted connections, the certificate is stored and verified on every subsequent check. Organization admins can view the pinned certificate, re-discover to detect a rotation, update the pin to a new certificate, or remove it entirely from the monitor detail page. See OPC UA.

July 8, 2026

New

- **Ask AI from the Query Builder** — The sparkle button in the Query Builder now opens the AI Canvas with a pre-filled prompt asking the assistant to help construct a query over your monitoring data, so you can go from exploring data to an AI-guided query in

one click. See Query.

Improved

- **AI Canvas cards flow in a single horizontal row** — Canvas cards are now laid out in a left-to-right filmstrip instead of stacked columns, making it easier to follow the conversation from one step to the next. See AI Canvas.
- **On the appliance, the TUI reloads automatically after an update or rollback** — After a successful system update or generation switch, the interface automatically restarts with the new version instead of remaining on the old one.

July 7, 2026

Improved

- **AI Canvas shows step order with numbered badges and connectors** — Canvas cards now display a sequence number in the corner and are connected by S-curve arrows in execution order; hovering a card highlights its connectors so you can trace the flow through a busy canvas. See AI Canvas.
- **AI Canvas run errors are now expandable** — When a run fails, a compact error banner appears at the top of the canvas; click it to expand the full error detail so the error no longer obscures the canvas. See AI Canvas.

Fixed

- **AI Canvas cards no longer overlap as content streams in** — The canvas layout now uses each card's real rendered height for masonry packing, so cards below a growing or flipping card shift down correctly instead of being overlapped. See AI Canvas.
- **On the appliance, the TUI header now shows the correct version after an update** — The version number in the header refreshes immediately when you return from the update or rollback screen, without needing to restart the TUI.

July 6, 2026

New

- **Ask AI from a CVE finding** — Each CVE finding detail page now has an "Ask AI" button that opens the AI Canvas with a prompt pre-filled to explain the vulnerability and analyze the practical risk for that specific device in your environment. See Vulnerabilities.

Improved

- **AI Canvas can query and display CVE findings** — The AI Canvas assistant can now search your organization's CVE findings and render them as structured cards, making it possible to ask vulnerability-focused questions alongside monitor and incident queries. See AI Canvas.

July 4, 2026

New

- **Vulnerabilities page** — A new Vulnerabilities section in the navigation lists all CVE findings across your devices, with a dashboard showing counts by severity and triage status, a detection-vs-resolved trend chart, the most-affected devices, and a searchable, filterable findings list with a detail page per finding. See Vulnerabilities.
- **Triage CVE findings** — Organization admins can now record a triage decision on each vulnerability (Open, In Progress, Resolved, Won't Fix, Not Affected, or Risk Accepted), add a written reason, and set a risk-acceptance expiry date, all from the finding detail page. See Vulnerabilities.
- **"Ask AI" on list pages** — The Monitors, Incidents, and Vulnerabilities list pages now each show an "Ask AI" button that opens the AI Canvas pre-filled with a status-report prompt for that section, ready for you to review and send.

Improved

- **Canvas prompt auto-grows and accepts pre-filled prompts** — The prompt textarea on the Canvas page now expands as you type, and list pages can deep-link to the canvas with a suggested prompt that you can review or edit before sending.
- **Canvas list handles deleted owners gracefully** — When a canvas creator has left the organization, their canvases now show "Deleted owner" in the list instead of a blank or broken name.

Fixed

- **CVE incident labels no longer show "undefined"** — Restored missing translation keys so older CVE-related incidents display the correct triage status labels instead of raw key strings.

July 3, 2026

Fixed

- **OPC UA certificate authentication now works with more servers** — Pulse's application instance certificate now includes the required extensions (ApplicationUri, key usages) mandated by the OPC UA specification, and correctly transmits both the certificate and its private key, resolving connection failures against mbedTLS-based servers such as open62541. See OPC UA.

June 30, 2026

New

- **AI Canvas** — A new AI Canvas page lets you ask questions about your incidents, monitors, or system state; the assistant lays out its findings as step cards on a spatial, pannable canvas. You can create multiple canvases, rename them (the AI auto-titles each one after the first run), revisit previous sessions from the canvas list, and delete canvases you no longer need. AI Canvas is available to organizations whose license includes the feature.

Improved

- **Wide tables in canvas cards scroll instead of overflowing** — Tables returned by the assistant now become horizontally scrollable when they are wider than a canvas card, so the card width stays predictable.

June 24, 2026

Fixed

- **On the appliance, backup restore and enrollment reset no longer prompt for a password** — Both operations now complete unattended; a missing absolute path for `systemctl` had caused them to stall waiting for a sudo password.

June 23, 2026

New

- **Filter monitors by muted state** — The monitor list now lets you show or hide muted monitors, so you can focus on the ones that are actively alerting. See Monitors.

Improved

- **Stale-data badge** — Monitors now display a badge when their latest reading has gone stale, making it obvious at a glance when a source has stopped reporting fresh values. See Monitors.

Fixed

- **Reliable incident actions** — Acknowledging or resolving an incident from a notification no longer submits twice when tapped in quick succession.

June 21, 2026

New

- **Vulnerability (CVE) monitoring** — Pulse can now match your connected devices against a known-vulnerability catalog and open an incident when a relevant CVE is found. Includes a dedicated CVE detail view, an on-demand re-scan, and CSV import for CVE monitors. See Monitors.
- **Manage extra certificate names on the appliance** — You can now add additional DNS names and IP addresses to the appliance's TLS certificate directly from the on-device menu, without rebuilding the appliance. See TLS Certificates.

Part 2 · Introduction

2.1 What is Pulse?

Pulse is a monitoring solution for operational technology (OT) infrastructure. It provides real-time visibility into industrial control systems, including OPC UA servers and Siemens S7 PLCs.

Key Capabilities

- **Data Sources** -- Connect to OPC UA servers, S7 PLCs, and Prometheus endpoints, and read their values continuously
- **Condition Monitoring** -- Alert on a reading, on a connection going quiet, or on a combination of both
- **Incident Management** -- Automatic incident creation when monitors detect problems, with acknowledgment and resolution workflows
- **Escalation Policies** -- Define multi-step notification rules with configurable severities, delays, and recipient teams
- **Team Management** -- Organize users into teams with role-based access control (Admin / Member)

Supported Protocols

OPC UA

Open Platform Communications Unified Architecture is a machine-to-machine communication protocol used in industrial automation. Pulse supports:

- Server connectivity, as a signal a condition can read
- Individual node values
- Security modes: None, Sign, Sign & Encrypt
- Authentication: Anonymous, Username/Password, Certificate

Siemens S7

The S7 communication protocol is used by Siemens SIMATIC PLCs. Pulse supports:

- PLC connectivity (IP, Rack, Slot), as a signal a condition can read
- Tag values in S7 address format (e.g., `DB1.DBW0`)
- S7-1200 and S7-1500 series

Architecture Overview

Pulse consists of several components:

- **Web Application** -- React-based dashboard for monitoring and configuration
- **Backend Server** -- Haskell-based API server handling business logic
- **Protocol Services** -- Dedicated Go services for OPC UA and S7 communication
- **Database** -- PostgreSQL for persistent storage of monitors, incidents, and configuration

2.2 Getting Started

> Setting up the Pulse appliance from the ISO image? Start with the **appliance installation guide** instead — this page covers the application-level setup that follows.

This guide walks you through the initial setup of Pulse after installation.

First Login

After deploying Pulse, open the web interface in your browser. You will see the login page.

If you don't have an account yet, click **Register** to create one.

Creating an Account

1. Navigate to the registration page
2. Enter your email address and choose a password
3. Your password must meet the following requirements:
   - Between 8 and 64 characters
   - At least one uppercase letter
   - At least one lowercase letter
   - At least one number
   - At least one special character

Forgot Your Password?

If you forget your password, click **Forgot password?** on the login page and enter your email address. If an account exists for that address, Pulse sends a password reset link to your inbox. The link expires in one hour.

Setting Up Your Organization

After registration, you will be prompted to create your organization. An organization is the top-level container for all your monitoring configuration.

1. Enter your organization name
2. Select your timezone

Next Steps

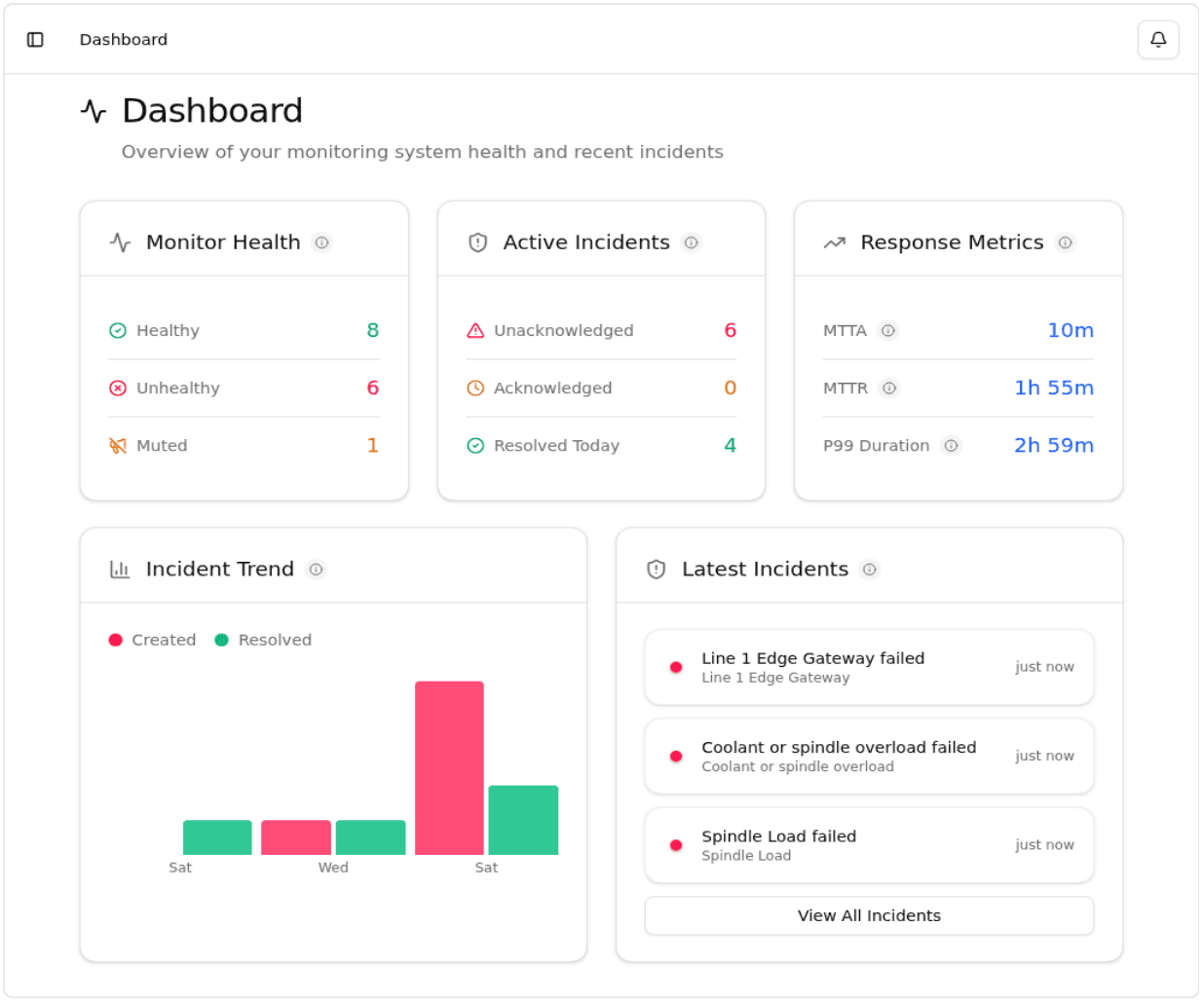
Once your organization is set up, you can:

- Configure your first monitor
- Set up teams
- Create escalation policies
- Customize your preferences

Part 3 · Monitoring

3.1 Dashboard

The dashboard provides a real-time overview of your entire monitoring infrastructure at a glance.



Monitor Health

The top section displays the current health status of all configured monitors:

- **Healthy** -- Monitors that are operating within expected parameters
- **Unhealthy** -- Monitors that have detected issues
- **Muted** -- Monitors that are temporarily silenced

Active Incidents

The incidents section shows the current state of open incidents:

- **Unacknowledged** -- New incidents that require attention
- **Acknowledged** -- Incidents that have been seen but not yet resolved
- **Resolved Today** -- Incidents that were resolved within the current day

Response Metrics

The dashboard tracks key incident response metrics over the last 7 days:

- **MTTA** -- Mean Time To Acknowledge: Average time from incident creation to first acknowledgment
- **MTTR** -- Mean Time To Resolve: Average time from incident creation to resolution
- **P99 Duration** -- 99th percentile incident duration

Incident Trend

A chart displays the trend of created versus resolved incidents over the past 7 days. This helps identify whether your team is keeping up with incoming incidents.

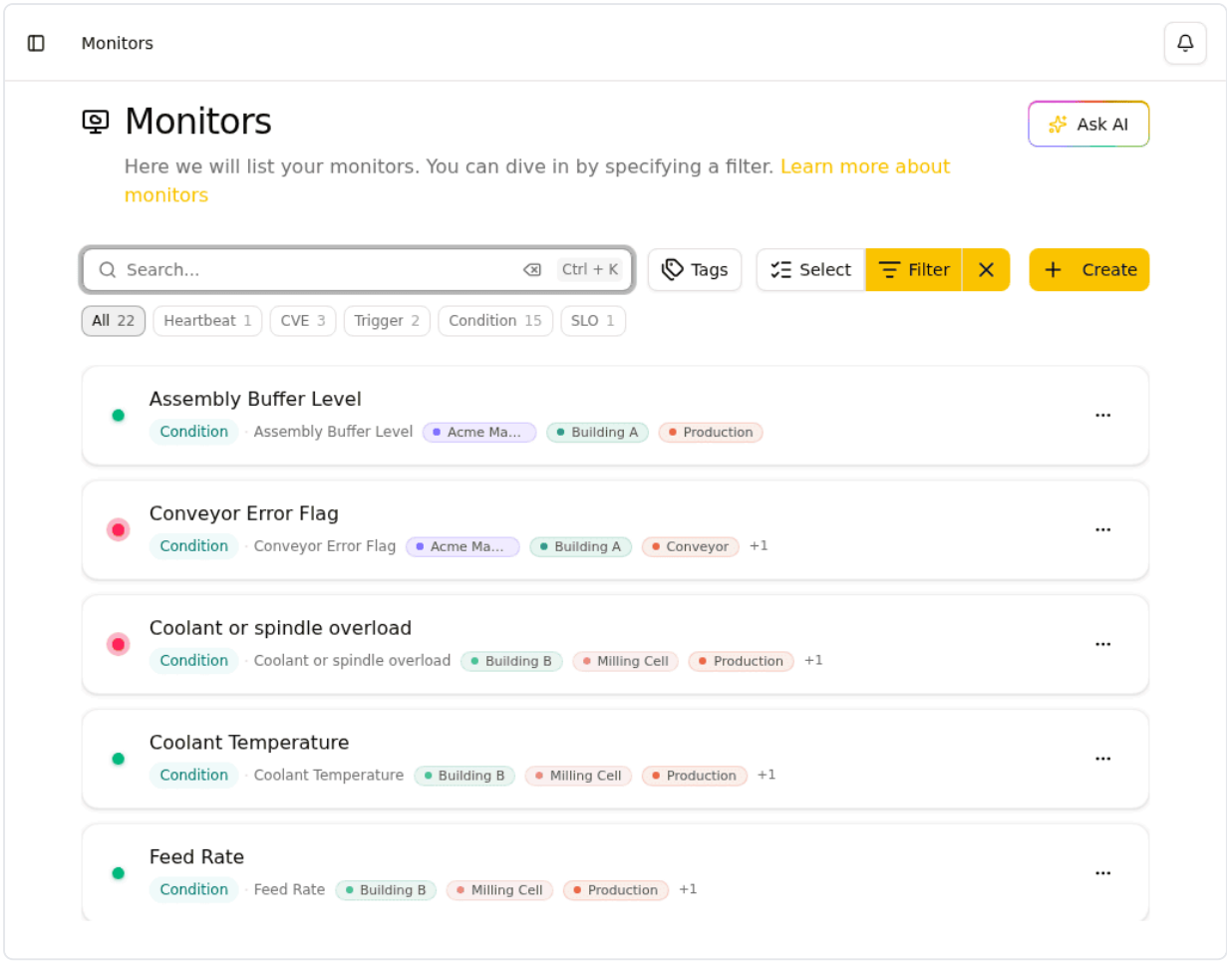
Latest Incidents

The bottom section shows the three most recent incidents with their current status, providing quick access to the most pressing issues.

Part 3 · Monitoring

3.2 Monitors

Monitors are the core building blocks of Pulse. Each monitor watches a specific aspect of your OT infrastructure and creates incidents when problems are detected.



Monitor Types

Pulse supports several types of monitors:

Condition Monitor

Evaluates a **condition** — a visual expression built from data-point readings, monitor states, and logic nodes — and raises an incident when the expression is true. This is the monitor type you use to alert on anything Pulse reads from an OPC UA server, an S7 PLC, or a Prometheus endpoint.

Creating a condition monitor:

Go to **Monitors → New monitor → Condition monitor**. The create page collects the monitor's own fields alongside the condition:

- **Name** -- Display name for the monitor
- **Escalation policy** -- Which policy is triggered when the condition fires
- **Tags** -- Optional tags for categorization
- **Condition** -- Described inline using a sentence form (for the common threshold case), or you can attach an existing condition with **Use an existing condition**

The sentence form covers the everyday case — a reading that crosses a line. Anything beyond what the sentence can express links out to the full condition editor, which is also where you go to make changes afterwards.

If the condition already exists (for example, you want a second monitor with a different escalation policy on the same condition), use **Use an existing condition** to select it. The condition stays read-only on this path — changes to it are made in the editor.

Editing an existing condition monitor:

Open the monitor's detail page and click **Edit**. The edit form shows the monitor's current escalation policy and tags; the condition itself is shown read-only and opens in the editor from a link.

A single condition can back multiple monitors, each with its own escalation policy. The condition is evaluated every five seconds. The monitor state is set to **Unhealthy** when the condition fires, **Healthy** when it clears, and **Unknown** when data is absent or a window is empty. An unknown result leaves any open incident open — Pulse does not treat "no data" as an all-clear.

Adding an OPC UA or S7 data source creates one of these for you, named *<source> unreachable*, unless you untick the box while adding the source. It is an ordinary condition monitor from the moment it exists.

CVE Monitor

Tracks published Common Vulnerabilities and Exposures (CVEs) that affect a specific device. Pulse scans the security advisory catalog regularly and surfaces each matching CVE as a **finding** in the Vulnerabilities section, where you can triage it.

- **Vendor** -- Device manufacturer (e.g., Siemens)
- **Order Code** -- Device order number / MLFB used to match advisories (e.g., `6ES7518-4AX01-0AB0`)
- **Firmware Version** -- Optional. Narrows matches to advisories that affect this specific firmware version.

A CVE monitor is shown as **unhealthy** while it has any finding that has not yet been triaged to a conclusive status. From the monitor's detail page, each matching finding links directly to its entry in the Vulnerabilities page via **View finding**.

> **INFO**
>
> CVE monitoring requires a license that includes CVE monitoring. If your license does not include this feature, the CVE Monitor create page shows an informational message. Contact your administrator to enable it.

Trigger Monitor

Receives inbound webhooks from external systems and raises incidents in Pulse. When you create a trigger monitor, Pulse generates a unique **ingest URL**. External systems POST any text or JSON payload to that URL; Pulse parses the payload with the rules you configure and creates, updates, acknowledges, or resolves incidents accordingly.

Basic configuration:

- **Name** -- Display name for the monitor
- **Escalation policy** -- Which escalation policy to use when an incident is raised

Payload parsing rules:

Each rule extracts a value or tests a condition from the incoming request body. Available modes are **Regex** (with optional capture group), **JSONPath**, and (for conditions only) **Contains text**.

- **Alert ID** -- Uniquely identifies an incident across multiple POSTs, enabling deduplication, acknowledgement, and resolution. When no Alert ID rule is set, every POST creates a new incident.
- **Sending system** -- Names the system a delivery came from, so that two senders sharing one ingest URL keep separate incidents even when they use the same Alert ID. Leave it unset if only one system posts to this monitor.
- **Source timestamp** -- Extracts the time the sending system says the alert fired (Alertmanager's `startsAt`, a ticketing webhook's `created`, and similar fields). Accepted formats are RFC 3339 with an explicit UTC offset, epoch seconds, and epoch milliseconds; anything else, including a value with no timezone, fails to parse. Shown on the incident timeline and in the delivery log -- see When the sender's clock disagrees with Pulse's below. Optional, and unrelated to Alert ID or Sending system.
- **Incident title** -- Extracted text used as the incident title. Falls back to the monitor name when not set or no match.
- **Incident description** -- Optional extracted text stored as the incident description.

- **Acknowledged when** -- A condition that, when matched, acknowledges an existing open incident with the same Alert ID. Requires Alert ID to be configured.
- **Resolved when** -- A condition that, when matched, resolves an existing open incident with the same Alert ID. Requires Alert ID to be configured. Resolve is evaluated before acknowledge.
- **Reset signal** -- A condition that, when matched, closes every open incident this monitor still holds for the sending system that posted it. Requires a Sending system rule. Reset is evaluated before resolve and acknowledge, and a delivery that matches it is never also read as an alert.

Several systems behind one URL

One ingest URL is often handed to more than one system -- two plants, two clusters, several alert groups. Those systems tend to reuse the same generic Alert IDs (`NodeDown`, `DiskFull`, an alert name with no site label), and without a **Sending system** rule Pulse cannot tell two such reports apart: they share one incident, either system's resolve closes it for both, and whoever is on call sees one incident where there are really two problems.

Configure **Sending system** to extract something that differs per sender -- a site label, a cluster name, the alert group key -- and each sender gets its own incidents. The value is shown on the incident and in the delivery log, so a copy-pasted configuration that leaves the same value at two sites is visible rather than silent.

Add the rule while the monitor is quiet. Incidents that are already open were recorded before Pulse could tell the senders apart, so they belong to no sender -- and once the rule is in place, neither a resolve nor a reset from a sending system will close them. The editor warns you when the monitor is holding such incidents; resolve those by hand.

Pulse takes the sending system from the payload as it is given. It does not verify it: anyone who can POST to the ingest URL can claim to be any sender, and therefore reset another sender's incidents. The URL is the boundary that matters, and the sending system is bookkeeping inside it. **If you need two systems to be unable to affect each other, give them separate trigger monitors** -- separate URLs are the isolation, not separate sending-system values.

Letting a system start over

A system that restarts, is redeployed, or is reconfigured can lose track of what it already reported. Its incidents then stay open in Pulse forever, and the monitor stays red, because nothing will ever arrive to resolve them. The **Reset signal** is how an integration says *"I have lost my state -- close what you still hold for me."*

A reset closes only the incidents belonging to the system that sent it, and only ones that were already open when it arrived. Two things it deliberately leaves alone:

- **Acknowledged incidents.** Somebody is already working on those, so a reset does not take them away; they stay yours to close.
- **Everything from other senders on the same monitor.**

A reset is not an all-clear. The incidents close because the evidence for them is gone, not because Pulse saw anything recover -- a condition that is still physically true on the plant floor will be closed by a reset from the system that reported it. Because of that, these closes are recorded as their own kind of event on the incident timeline rather than looking like an ordinary resolution, and the delivery log shows how many incidents each reset closed.

> **Configure a reopen window alongside reset**
>
> A system that resets almost always re-posts whatever is still wrong immediately afterwards -- that is the whole point of a restart. What happens next depends on the **reopen window** of the escalation policy the monitor uses, which is **zero unless you set it**.
>
> With a reopen window, each re-posted alert reopens the incident it closed moments earlier: the same incident, the same responder, the same position in the escalation, and nobody is notified again. Without one, the closes are final and every re-post starts a fresh incident that notifies from the beginning -- a reset of forty alerts becomes forty new pages.
>
> The monitor editor warns you when a reset rule is configured on a policy with no reopen window. See Escalation Policies.

When the sender's clock disagrees with Pulse's

Pulse reports on when it *heard about* an alert, not on when the sender says it happened. Escalation timing, SLAs, incident durations, and every report all run on the moment the delivery arrived -- Pulse's own clock, always. A **Source timestamp** rule does not change that; it adds the sender's claim as extra evidence alongside it, never in place of it.

When configured, the value appears on the incident timeline labelled **Reported by sender**, next to the delta to when Pulse actually received it -- for example, "Reported by sender: 14:03, 4 minutes before Pulse received it." If a later delivery for the same alert carries a different value than the one already recorded, that change gets its own timeline entry, so a sender correcting itself or drifting mid-incident is visible rather than silently overwritten.

A large or negative delta -- the sender claiming a time *after* Pulse's own receipt -- almost always means the sending system's clock is wrong, not that Pulse missed something. Pulse shows that delta as-is rather than hiding or clamping it, because catching a broken sender clock is the point of showing the value at all.

The rule only ever affects the incident it raises or reopens; a value that is absent or fails to parse simply leaves the alert untouched with no source timestamp recorded -- it never blocks the delivery the way a misconfigured Alert ID rule can.

Request headers:

You can require one or more HTTP headers on every inbound POST. All configured headers must match for Pulse to accept the request — a missing or non-matching header returns a 401. This lets you put your own shared-secret authentication in front of the ingest URL (e.g., require `X-API-Key: your-secret`). Match modes are **Equals**, **Contains text**, and **Regex**.

Ingest URL and delivery log:

The monitor's detail page shows the ingest URL with **Copy** and **Rotate URL** buttons. A **Last delivery** timestamp shows when the most recent webhook was received; if nothing has arrived yet it reads *No delivery received yet*. Rotating the URL immediately invalidates the old one — update any systems that POST to it.

Incoming webhook deliveries appear in the activity feed alongside state-change events. Each delivery entry shows the outcome (incident created, updated, acknowledged, resolved, ignored, incident reopened, or reset), the extracted Alert ID, the sending system where one was read, and a **Details** button. A reset entry shows how many incidents it closed instead of linking to a single one. Opening **Details** shows the full payload as it arrived, the content type, a formatted view where applicable, and a direct link to the associated incident. If several deliveries arrive in quick succession they are collapsed into a single entry — click **Show all N** to expand the group. A notice at the bottom of the feed marks the point beyond which older deliveries are no longer retained.

> **TIP**
>
> Treat the ingest URL as a secret. Anyone who has it can POST to your monitor. Use request-header matchers to add an extra layer of authentication.

Heartbeat Monitor

Watches a job that runs on a schedule -- a nightly backup, an hourly export, an edge device that reports in. The job calls a secret **ping URL** every time it finishes successfully, and Pulse raises an incident when the ping stops arriving. This is the only monitor type where Pulse waits instead of connecting, which makes it the right choice for anything Pulse cannot reach directly.

- **Name** -- Display name for the monitor, usually the job's name
- **Expected interval** -- How long Pulse waits for a ping before raising an incident (minimum 1 minute)

- **Escalation policy** -- Which escalation policy to use when the ping stops

The ping URL is a secret: anyone who has it can fake a heartbeat. The ping must run **after** the job and only when the job succeeded. See Heartbeat Monitors for the full walkthrough, including how to wire it into cron without silently defeating the monitor.

SLO Monitor

An objective is itself a monitor: it carries an escalation policy, tags and a mute switch, and pages somebody when its error budget is burning too fast. It lives in the **Objectives** section rather than in this list — the burn-alert ladder sits beside the promise it watches. See Objectives → Being told about it for the full setup guide.

Conditions

Conditions are reusable graphs that express the logic for when a monitor should alert or measure a promise. Each condition is a directed acyclic graph of typed nodes — data points, aggregations, comparisons, logic gates, and one or more output nodes — that Pulse evaluates against live data.

The **Conditions** entry at the bottom of the sidebar, beside Tags and Severities, opens the conditions list at `/conditions`. It sits there rather than under Monitors because a condition is mostly the monitor's editing surface — you reach one by opening the monitor that alerts on it; the library is for the case where several monitors, or an objective, share the same condition.

Each row in the list shows:

- A **status dot** reflecting the condition's current verdict — **Not holding** (red), **Holding** (green), **Unknown**, **Stale** (not being evaluated), or **Never evaluated**
- When the condition was last updated
- How many **data points**, **monitors**, and **objectives** use it
- An **Actions** menu (the `···` button) with shortcuts to open the condition, jump to the monitors that alert on it, open the objectives about it, or delete it

Use the **State** filter to narrow the list to conditions in a specific verdict, or the **Used by nothing** toggle to find conditions that no monitor or objective currently references. Administrators can create a new condition with **New condition** or delete an existing one. A condition cannot be deleted while a monitor is alerting on it or an objective is promising about it; remove those first.

Deleting several conditions at once

Organization administrators can clear out a run of conditions in one gesture. Bulk selection is available on desktop only.

Click **Select** in the toolbar to enter selection mode. Every row then works as a checkbox and the whole row is the target — clicking a row selects that condition instead of opening it. A **Select all N conditions** checkbox above the list ticks or clears every row matching the current filter, and holding **Shift** while clicking a row adds every row between the one you last clicked and this one. Changing the filters clears the selection. To leave the mode, click **Select** again or press **Escape** — the first press clears the selection, a second leaves the mode.

With at least one condition selected, a floating bar appears at the bottom of the list with **Delete** and a **×** that clears the selection. Delete opens a confirmation dialog that asks you to type the number of conditions you are about to remove.

The same rule as a single delete applies, and it applies to the whole selection at once: a condition a monitor alerts on, or an objective promises about, cannot be deleted. If any selected condition is still in use, nothing at all is deleted — the dialog then lists which conditions are held and by what, and offers to delete the rest of the selection instead. Combining the **Used by nothing** toggle with **Select all** is the quickest way to clear away drafts and leftovers, since nothing in that list can be blocked.

Clicking a condition's name opens its graph in the condition editor. From the editor you can:

- Build or modify the node graph on the canvas
- Preview how the condition would have fired over a past time range (**Preview** mode)
- See the appliance's current live evaluation of the stored graph (**Live** mode)
- Save the condition and optionally attach or update a monitor

Condition editor layout

The editor has three main areas arranged left to right: the **Palette** (node types you drag onto the canvas), the **canvas** itself, and the **Inspector** (properties and issues for the selected node).

You can collapse either side panel to give the canvas more room. Click the toggle at the edge of the Palette or Inspector to fold it down to a narrow strip; click the strip again to restore it. The editor remembers which panels you had open or closed, so your layout is restored when you return — this is stored per browser and does not follow you to another machine.

The **preview panel** at the bottom shows data point charts for the time range you select. You can drag its top edge to resize it. The height you set is also remembered per browser and clamped to fit the window whenever you open the editor on a different screen.

Preview vs Live — A toggle in the timeline panel switches between two value sources:

- **Preview** — Backtests the current draft over the selected time range. This is the default; it answers "would my change fire?".
- **Live** — Reads the appliance's current evaluation of the *saved* graph in real time. Use this to see what the condition is actually doing right now. Live values are not available for an unsaved condition or while the draft has unsaved changes; save first.

When Live is active, a badge in the top-right of the canvas labels the node values as live so they are not mistaken for backtest results.

Condition editor on a phone — The editor is a desktop workspace and is not available on phones. Opening a condition editor URL on a narrow viewport shows a **read-only** view instead: a **Graph** tab with the condition canvas (pan and zoom, no editing) and a **Details** tab with the current live evaluation of each node. Creating a new condition requires a desktop browser.

Unreadable saved graph — If the stored graph of a condition cannot be opened (for example, because it was saved by a newer version of Pulse), the editor shows a banner explaining this. You can draw a new graph on the empty canvas and save it to replace the unreadable one.

Keyboard shortcuts — The canvas responds to keyboard shortcuts for selecting, editing, and navigating nodes. Press `?` or click the keyboard icon in the canvas controls to open the shortcuts reference panel, which lists every gesture the canvas answers to.

Condition node types

The condition editor palette groups nodes into five categories:

Alerts & objectives — output nodes that give the graph meaning:

Node	What it does
Alert	Raises an incident when its input is true. A condition can carry several named Alert nodes, each raising its own separately-named incident
Objective	Measures how much of the time its input holds, as a service level objective. Connect it to the same signal as the Alert node — or its inverse — so the promise and the page share one definition of "good"

Sources — values that enter the graph:

Node	What it outputs
Sensor	The live reading from a data point on one of your data sources, or that source's own reachability
Learned normal	The measured baseline statistic for a data point (see below)
Monitor state	True when a monitor is healthy or unhealthy
Constant	A fixed number or boolean

Transform — numeric operations:

Node	What it outputs
Aggregate	Min, max, average, sum, or count over a time window
Change over time	How much a data point's value has changed within a time window
Rate of increase	How fast a counter is climbing, per second, treating any drop as a counter reset
Math	Arithmetic on two numbers
Compare	True when a numeric comparison holds

Chained aggregate windows add up

Each Aggregate node's window must stay within Pulse's hot-tier data retention period. When you chain aggregates (for example, an average over another average), the windows accumulate: an outer 24-hour average over an inner 24-hour average asks Pulse to read 48 hours of history. The editor flags the deepest node where the cumulative path first crosses the retention limit — shorten one of the windows in the chain to fix it.

Logic — boolean operations:

Node	What it outputs
Gate (AND / OR)	True when all or any inputs are true

Node	What it outputs
Not	Inverts a boolean
Sustained	True only after its input has held for a specified duration — use **Continuously true** to require the input stays true throughout the window, or **True at any point** to require it was true at least once

Resilience — handling absent data:

Node	What it outputs
Fallback	Substitutes a default when its input has no data
Has data	True when its input currently has data

Condition node roles

Certain conditions — particularly those generated automatically by Pulse for anomaly-detection monitors — tag specific nodes with a **role**. A role identifies which node's value the incident description will quote by name when the condition fires (for example, "78.2 is above the upper limit of 76.0").

Three roles exist:

Role	What it marks
Observed	The data point reading being compared
Upper limit	The computed upper bound
Lower limit	The computed lower bound

A tagged node shows its role as a prefix in the card's summary line — for example, **Upper limit · avg over 15 m**. Selecting the node opens the inspector, which displays a **Quoted in incidents as** note confirming which role it carries. The role is set by Pulse when generating the condition; the editor carries it but does not let you change it.

Learned normal (baseline)

The **Learned normal** node reads the measured baseline for a specific data point — what Pulse determined the data point normally does over the last 14 days. This makes it possible to write conditions like "the reading is more than two normal spreads above the data point's normal level" without knowing the exact numbers upfront.

Each baseline node exposes one statistic at a time:

Statistic	Meaning
Normal level	The data point's typical value (median)
Normal spread	How far the data point typically strays from its normal level
Normal low (1%)	The bottom of the data point's normal range
Normal high (99%)	The top of the data point's normal range
Normal movement	How much the data point typically moves within a few minutes
High movement	The upper edge of the data point's typical movement
Normal data coverage	The fraction of the window in which the data point normally has data

The baseline is computed by the nightly measurement job described in Anomaly Detection. A data point added today will not have a baseline until the next nightly run; a node pointing at it reports **Unknown** until then.

Reading Quality

A condition monitor reports **Unknown** rather than Healthy when the readings it needs are missing or unreliable — for example because the PLC connection dropped or a value is no longer available on the server. An unknown verdict leaves an open incident open and pauses escalation; it resolves itself as soon as a valid reading arrives.

Where a reading has gone quiet is visible on the data source: its detail page shows each data point's last value, the time it was last read, and whether the last reading was Good, Bad, or No data. The source list shows the same at a glance — **Reading**, **Not reading**, or **Awaiting first read**.

Filtering Monitors

The monitor list supports several filter options:

- **Status** -- Filter by Healthy, Unhealthy, or Unknown
- **Type** -- Filter by monitor type (Heartbeat, CVE, Trigger, Condition)
- **Tags** -- Filter by custom tags assigned to monitors
- **Escalation Policy** -- Filter by the escalation policy assigned to the monitor
- **Data source** -- Filter by which data source the monitor reads from

- **Stale data** -- Show only monitors whose last reading is stale
- **Muted only** -- Show only monitors that are currently muted
- **Search** -- Text search across monitor names

A **type chip row** below the search bar shows each monitor type alongside its count. Click a chip to filter by that type; click it again to clear. A **Tags** button opens the tag picker directly — tags are the main way to slice the list, so they sit in the toolbar rather than inside the filter menu.

The sidebar's **Monitors** entry opens this list unfiltered. Under it sit shortcuts for the two types that behave differently enough to be worth their own entry: CVE Monitors and Trigger Monitors. Heartbeat and condition monitors have no shortcut of their own — they are the bulk of the list, found by tag or by the **Type** filter. Objectives are in the **Objectives** section; they are monitors underneath but live beside their burn-alert ladders, not here.

OPC UA and S7 no longer appear here. A connection is a data source now, so those entries moved to **Data sources → OPC UA** and **Data sources → S7**, which is also where the old monitor-list links take you.

On mobile, the filter controls are replaced by a **Filter** button that opens a bottom drawer. The drawer has sections for Status, Type, and Tags, plus standalone toggles for **Stale data** and **Muted only**. Tap **Apply** to activate the selected filters or **Reset** to clear all draft selections. Active filters appear as read-only chips below the search bar; to change a selection, open the drawer again.

Bulk Actions

Organization administrators can act on several monitors at once from the monitor list. Bulk selection is available on desktop only.

Click **Select** in the toolbar to enter selection mode. In the mode every row carries a checkbox and the whole row is the target — clicking a row selects that monitor instead of opening it. A **Select all N monitors** checkbox above the list selects or clears every row that matches the current filter, and holding **Shift** while clicking a row adds every row between the one you last clicked and this one, without dropping anything already ticked. Changing the filters clears the selection, because everything selected has to be a row still in front of you.

To leave selection mode, click **Select** again or press **Escape**. Escape unwinds one step at a time: the first press clears the selection, a second leaves the mode.

Once at least one monitor is selected, a floating action bar appears at the bottom of the list:

- **N selected** — how many monitors are currently selected
- **Mute** — mutes all selected monitors. A toast confirms how many changed state and offers **Undo**, which reverses only the monitors the action actually changed, so a monitor you had already muted stays muted.
- **Unmute** — unmutes all selected monitors, with the same toast and undo.
- **Delete** — opens the bulk delete confirmation dialog.
- **×** (Clear selection) — clears the selection without leaving the mode.

Bulk delete

Deleting monitors in bulk opens a confirmation dialog that counts the true number of monitors that will be removed. Selecting an S7 Endpoint or OPC UA Endpoint monitor also deletes all of its child monitors (S7 Tag or OPC UA Node monitors for that endpoint), so the dialog fetches those child counts before asking for confirmation. Type the displayed number to unlock the delete button.

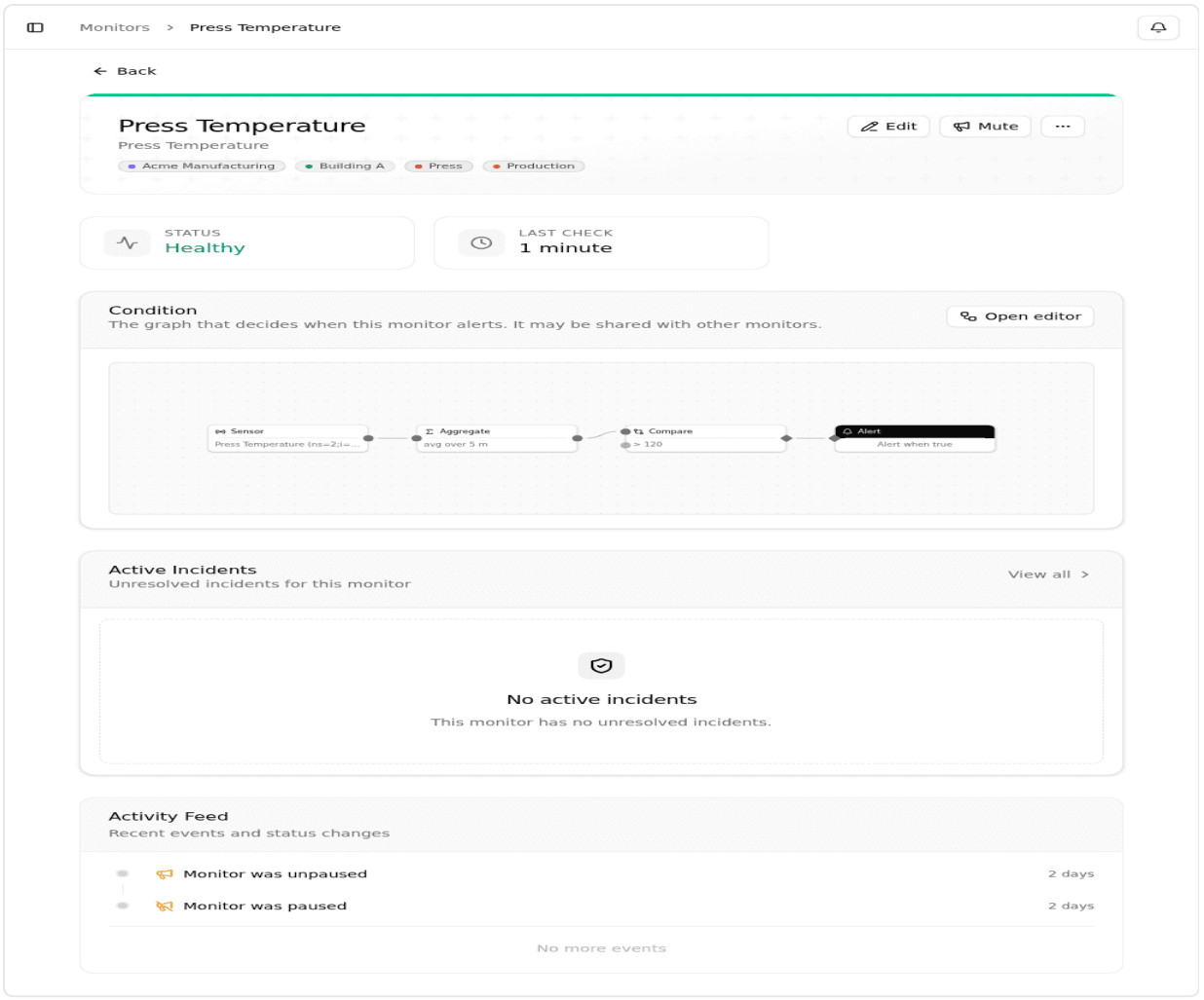
If any selected monitors are used by conditions, the delete is blocked and the dialog lists which conditions reference them. You can remove those monitors from their conditions first and retry, or proceed immediately to delete the remaining unblocked monitors.

A monitor and the condition it alerts on go together: deleting the last monitor that alerts on a condition deletes that condition too, since nothing else would be able to reach it. The toast reports both numbers — "Deleted 340 monitors — 340 conditions went with them" — so a cleanup after an upgrade does not leave you hunting for graphs that are already gone. A condition several monitors share, or one an objective promises about, stays where it is.

Conditions that nothing uses any more are cleared from the conditions list itself; see Conditions.

Monitor Details

Click on any monitor to view its detail page.



Every detail page shows:

- **Status** -- Current health status (Healthy or Unhealthy)
- **Configuration** -- What the monitor alerts on. For a condition monitor this is the condition itself, drawn as its graph; open it to edit. Because a condition can be shared by several monitors, it is edited where it lives rather than on this page.
- **Tags** -- Tags assigned to the monitor for categorization
- **Actions** -- Edit, mute, or delete the monitor via the Actions button
- **Activity Feed** -- a chronological history of status changes and events

Every monitor type has the Activity Feed, condition monitors included. When several monitors share one condition, each still has its own feed: they record the same status changes at the same moments, because that is what happened to each of them, and each monitor's own muting and unmuting appears only in its own feed.

Importing Monitors from CSV

Organization administrators can create CVE monitors in bulk from a CSV file. See Importing Monitors from CSV for the full walkthrough and the complete column reference.

Tags from Data Sources

A monitor carries its own tags, and you can filter the monitor list by them. Tags set on a **data source** are separate: they describe the machine rather than the alert, and every data point of that source inherits them automatically. Any condition monitor that reads one of those data points can be filtered by the inherited tags as well, so tagging a PLC once is enough to find everything that watches it.

Muting Monitors

You can temporarily mute a monitor to suppress incident creation during planned maintenance. Muted monitors continue to check their targets but will not generate new incidents.

> **TIP**
>
> Muting is useful during planned maintenance windows. Remember to unmute monitors after maintenance is complete.

Deleting Monitors

To delete a monitor, open its detail page, click **Actions**, and choose **Delete**. The confirmation dialog requires you to type the monitor's name before the delete button becomes active — this deliberate step prevents accidental loss of measurement history.

Deleting a monitor that is used by a condition is blocked. Remove the monitor from the condition first, then delete it.

3.3 Import from CSV

The same three-step wizard imports three different things, depending on where you open it:

- From the monitor list's **Create** dropdown, it creates **CVE monitors**. See Importing CVE monitors below.
- From an **S7 data source**'s **Import** action, it declares **data points** on that source. See Importing S7 data points below.
- From an **OPC UA data source**'s **Import** action, it declares **data points** on that source. See Importing OPC UA data points below.

The wizard guides you through the same three steps either way: Upload, Map columns, and Results — only the column set and what a row builds differ.

Importing CVE monitors

Organization administrators can create many **CVE monitors** at once from a CSV file. In the monitor list, open the **Create** dropdown and choose **Import CSV** to open the import wizard.

CVE monitors are the only monitor type this import creates. Connections to OPC UA servers and S7 PLCs are added as data sources, and everything that alerts on a value is a condition monitor — neither is a row in a spreadsheet. To declare S7 tags or OPC UA nodes in bulk, see Importing S7 data points or Importing OPC UA data points.

Two values behave in ways worth knowing up front:

- `escalationPolicy` references an existing escalation policy by **name or numeric id**. A name must match exactly, so create the policy before importing.
- `incidentMessageTemplate` is the literal incident message, not a reference to a saved template.

The column set is listed under Map columns.

Importing S7 data points

Organization administrators can declare many **S7 data points** at once from a CSV of tag addresses. Open an **S7 data source**'s detail page and choose **Import** to open the wizard, already bound to that source.

Because the source is implied by where you started the import, the file is about tags and nothing else — no `endpoint`, `rack` or `slot` column, and no `monitorType` either. The column set is listed under Map columns.

A data point this import declares is **adopted** and reads on the source's own cadence, like any other data point of the source. A row naming an address and data type already declared on the source is a no-op — it resolves to the existing data point rather than creating a duplicate, so re-running an unchanged file is harmless.

Unlike the CVE import, this run is **all or nothing**: every row is checked before anything is written, and if any row fails, nothing is created — the results step lists every failing row so you can fix the file and try again. See Results.

Alerting on the values you import

Mapping four more columns — `aggregation`, `operation`, `operand` and `duration` — turns each row from a data point declaration into a threshold alert on it: "alert when the average of `DB1.DBD10` over 5 minutes is above 80". For a row with all four mapped, the import creates:

- a **condition** — the graph `Sensor → Aggregate → Compare → Alert` — pinned to the data point the row declares, and
- a **condition monitor** that alerts on it, in the **Conditions** group of the monitor list.

The condition opens in the same guided sentence form a hand-built threshold uses, so a hundred imported alerts stay editable afterwards — nothing about them is canvas-only.

Once any of the four is mapped, all four become required, along with `name` (the condition and monitor's name) and `escalationPolicy` (who an incident from it pages) — a row cannot build a threshold from three fields and a guess at the fourth. Leaving all four unmapped keeps the plain behavior above: rows declare data points only, with no conditions or monitors created.

Condition names are **unique per organization and case-insensitive**. A row whose name is already taken — by an existing condition, or by another row in the same file — fails validation before anything is written, the same way an invalid address does.

The **Activate created monitors** toggle on the mapping step decides whether the monitors this run creates start active or inactive. Off (the default) creates them **inactive**: they sit in the monitor list for review, and the evaluator skips an inactive monitor entirely, so none of them can raise an incident until you turn them on by hand. On starts them alerting immediately. This is a good default for a first import of a hundred thresholds — review the generated conditions, then activate the ones that look right.

Importing OPC UA data points

Organization administrators can declare many **OPC UA data points** at once from a CSV of node ids — the same case a machine builder's node list or a large server you'd rather not browse covers. Open an **OPC UA data source**'s detail page and choose **Import** to open the wizard, already bound to that source.

Because the source is implied by where you started the import, the file is about nodes and nothing else: a row is one column, `nodeId`. Unlike an S7 tag, an OPC UA data point's value type is not declared — it is learned from the first reading, the same as when you declare one by hand — so there is no `dataType` column to map. The column set is listed under Map columns.

Node ids are checked for valid **syntax** only — `ns=2;s=Line1.Temperature` or `ns=3;i=1001`, the same textual form the source detail page uses. Whether the node actually exists on the server is not checked at import time: that would make the import depend on a reachable server, which defeats the point of preparing a file before commissioning. A node id the server does not recognize comes back as a bad reading once the source starts sampling it, visible on the source's detail page like any other bad reading.

A row naming the source's own health-check node is refused, the same way declaring it by hand is — the point already exists as the source's reachability probe, so importing it again would create a second, meaningless liveness point rather than a real reading.

A data point this import declares is **adopted** and reads on the source's own cadence, like any other data point of the source. A row naming a node already declared on the source is a no-op — it resolves to the existing data point rather than creating a duplicate, so re-running an unchanged file is harmless, and two rows that spell the same node differently (say, with or without a leading zero in the identifier) still resolve to one data point.

Unlike the CVE import, this run is **all or nothing**: every row is checked before anything is written, and if any row fails, nothing is created — the results step lists every failing row so you can fix the file and try again. See Results.

Alerting on the values you import

Mapping four more columns — `aggregation`, `operation`, `operand` and `duration` — turns each row from a data point declaration into a threshold alert on it: "alert when the average of `ns=3;s=Boiler.Pressure` over 5 minutes is above 80". For a row with all four mapped, the import creates:

- a **condition** — the graph `Sensor → Aggregate → Compare → Alert` — pinned to the data point the row declares, and
- a **condition monitor** that alerts on it, in the **Conditions** group of the monitor list.

The condition opens in the same guided sentence form a hand-built threshold uses, so a hundred imported alerts stay editable afterwards — nothing about them is canvas-only. Because an OPC UA data point's value type is only learned from its first reading, a threshold over a freshly declared node imports and validates cleanly even if the node turns out to hold, say, a boolean — it simply won't fire sensibly under a numeric comparison like `avg ... > 80`.

Once any of the four is mapped, all four become required, along with `name` (the condition and monitor's name) and `escalationPolicy` (who an incident from it pages) — a row cannot build a threshold from three fields and a guess at the fourth. Leaving all four unmapped keeps the plain behavior above: rows declare data points only, with no conditions or monitors created.

Condition names are **unique per organization and case-insensitive**. A row whose name is already taken — by an existing condition, or by another row in the same file — fails validation before anything is written, the same way an invalid node id does.

The **Activate created monitors** toggle on the mapping step decides whether the monitors this run creates start active or inactive. Off (the default) creates them **inactive**: they sit in the monitor list for review, and the evaluator skips an inactive monitor entirely, so none of them can raise an incident until you turn them on by hand. On starts them alerting immediately. This is a good default for a first import of a hundred thresholds — review the generated conditions, then activate the ones that look right.

CSV format

The first row must be a **header row** of column names — the next step matches those names to fields (you can adjust the mapping by hand). Save the file as **UTF-8** and follow standard CSV rules (RFC 4180): separate values with commas, and wrap any value that itself contains a comma in double quotes.

To start from a working example, use **Download example CSV** on the upload step: it contains the columns for the wizard you opened, with one or more example rows.

Step 1 — Upload

Drag a CSV file onto the drop zone, or click to browse. A preview table shows the first rows so you can confirm the file was read correctly; use **Replace** to swap in a different file. Click **Next** to continue, or **Cancel** to close the wizard.

Step 2 — Map columns

Map each field to a column from your CSV. The wizard auto-detects matching column names and pre-fills the mapping where it can. For every field you can:

- pick the **CSV column** whose values to use,
- choose **Fixed value…** to apply one value to every row (handy for shared settings such as the escalation policy), or
- leave it **Not mapped** — allowed only for optional fields.

The table lists every column, whether it is required, and the values it accepts.

CVE monitor columns

Column	Required	Valid values / format
`monitorType`	Yes	`cve`
`name`	Yes	Free text
`escalationPolicy`	Yes	**Name** (exact match) or numeric **id** of an existing escalation policy
`vendor`	Yes	Device vendor, e.g. `Siemens`
`orderCode`	Yes	Device order code / MLFB, e.g. `6ES7516-3AN00-0AB0`
`firmware`	Optional	Firmware version, e.g. `V2.9.2` (empty → matches all firmware versions)
`incidentMessageTemplate`	Optional	Free text (the message itself)

> **TIP**
>
> **CVE monitors** match published advisories against a device by `vendor` + `orderCode` (+ optional `firmware`). See CVE monitors for how matching works.

S7 data point columns

Column	Required	Valid values / format
`address`	Yes	An S7 address, e.g. `DB1.DBW0` or `DB2.DBD4`

Column	Required	Valid values / format
`dataType`	Yes	`BOOL`, `BYTE`, `WORD`, `DWORD`, `INT`, `DINT`, or `REAL`
`name`	Only if any threshold column is mapped	What to call the condition and its monitor
`escalationPolicy`	Only if any threshold column is mapped	**Name** (exact match) or numeric **id** of an existing escalation policy
`aggregation`	All four, or none (see below)	`min`, `max`, `sum`, `avg` (or `average`), `count`
`operation`	All four, or none	`gt` (or `>`, `above`), `gte` (`>=`), `lt` (`<`, `below`), `lte` (`<=`), `eq` (`=`), `neq` (`!=`)
`operand`	All four, or none	The threshold value, e.g. `80`
`duration`	All four, or none	The aggregation window: seconds, or a number with a unit, e.g. `300`, `5m`, `1h`

TIP

See S7 data sources for what each address form and data type means, and Alerting on the values you import for the four threshold columns.

OPC UA data point columns

Column	Required	Valid values / format
`nodeId`	Yes	An OPC UA node id, e.g. `ns=2;s=Line1.Temperature` or `ns=3;i=1001`
`name`	Only if any threshold column is mapped	What to call the condition and its monitor
`escalationPolicy`	Only if any threshold column is mapped	**Name** (exact match) or numeric **id** of an existing escalation policy
`aggregation`	All four, or none (see below)	`min`, `max`, `sum`, `avg` (or `average`), `count`

Column	Required	Valid values / format
`operation`	All four, or none	`gt` (or `>`, `above`), `gte` (`>=`), `lt` (`<`, `below`), `lte` (`<=`), `eq` (`=`), `neq` (`!=`)
`operand`	All four, or none	The threshold value, e.g. `80`
`duration`	All four, or none	The aggregation window: seconds, or a number with a unit, e.g. `300`, `5m`, `1h`

> **TIP**
>
> See OPC UA data sources for the node id syntax and how a data point's value type is learned, and Alerting on the values you import for the four threshold columns.

When every required field is mapped, click **Run Import**. To close and discard your work, click **Cancel**; to pause without losing your column mappings, use the × button in the header — the wizard resumes where you left off when you reopen it.

Step 3 — Results

A summary shows how many rows were imported successfully and how many failed.

- **CVE monitors** commit row by row, so a failed row does not stop the rows around it from being created.
- **S7 and OPC UA data points** commit all together: if any row failed, the summary shows zero successes, and nothing was written.

Each failed row is listed with the error that caused it. Click **Download failed rows** to export just those rows as a CSV, correct them, and import the file again.

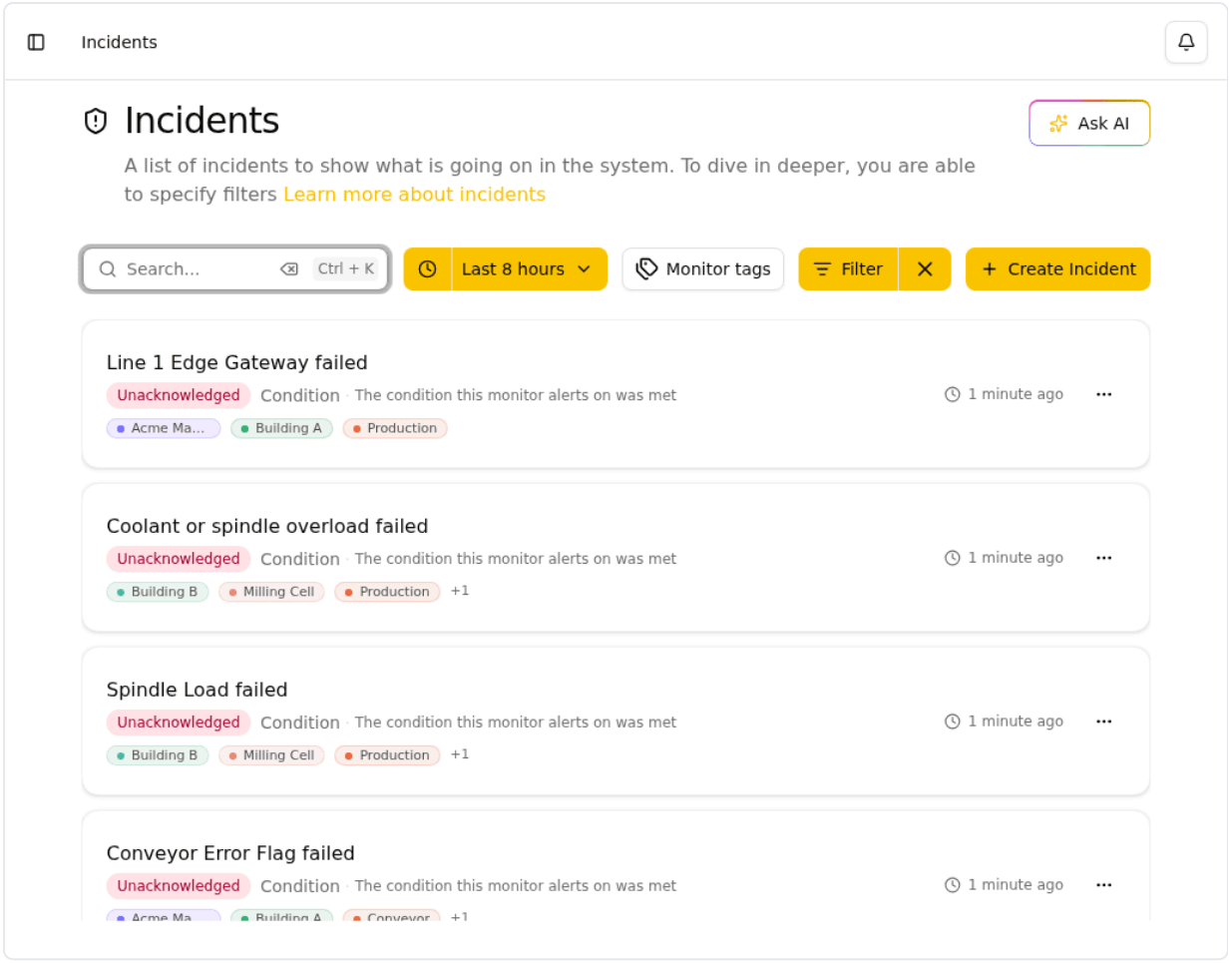
Related

- Monitors — monitor types, configuration, and evaluation
- S7 data sources — declaring data points by hand, adoption, and cadence
- OPC UA configuration — declaring data points by hand, adoption, and cadence

Part 3 · Monitoring

3.4 Incidents

Incidents represent a period of degraded or failed service that requires attention. They are created automatically when a monitor detects a problem, or manually by a team member.



Incident Lifecycle

Every incident follows a defined lifecycle:

1. **Unacknowledged** -- The incident has been detected but no one has responded yet. Shown with a red **Unacknowledged** badge.
2. **Acknowledged** -- Someone has seen the incident and is working on it. Shown with an amber **Acknowledged** badge.
3. **Resolved** -- The underlying issue has been fixed. Shown with a green **Resolved** badge.

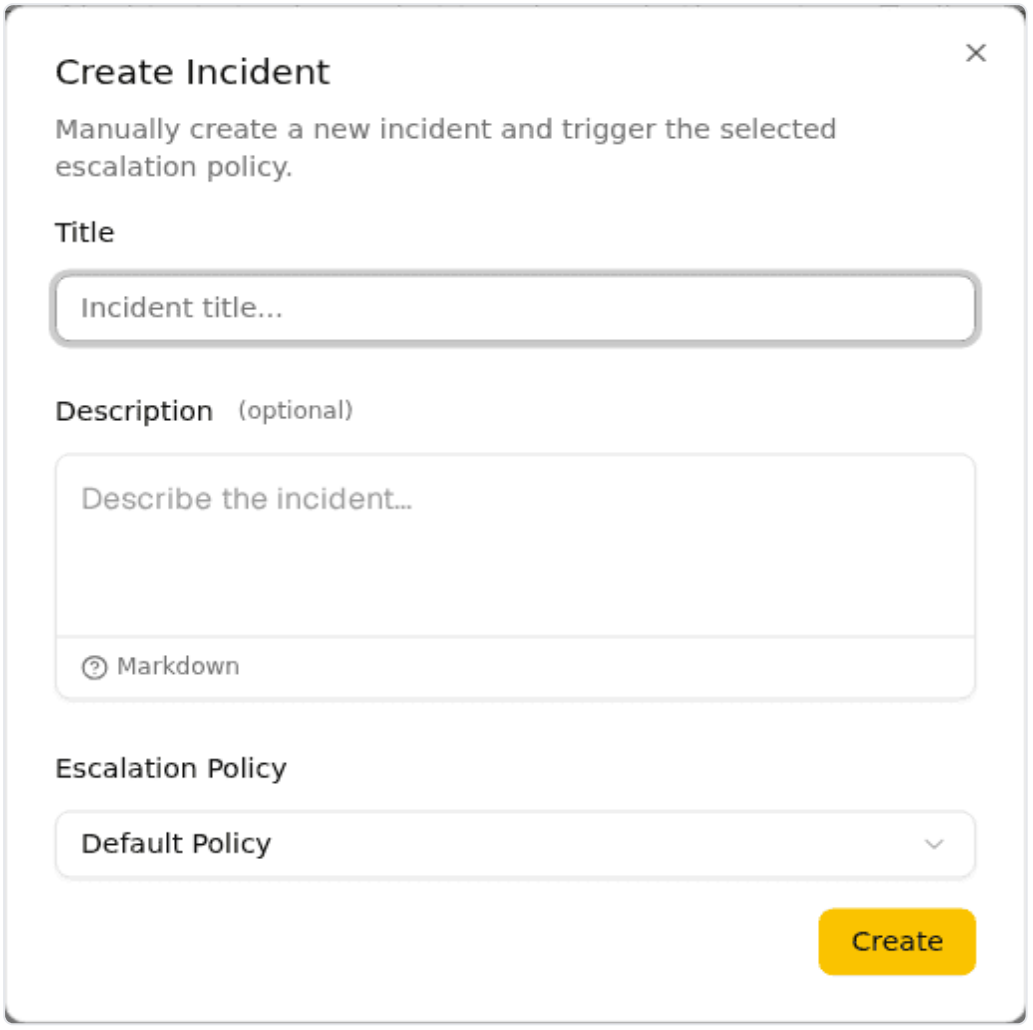
Creating an Incident Manually

You can manually create an incident from the incident list page. This is useful when you need to trigger an escalation policy for a situation not detected by a monitor.

To create a manual incident:

1. Open the **Incidents** page.
2. Click the **Create Incident** button in the toolbar. On mobile, this button appears as a floating action button fixed at the bottom-right corner of the screen.
3. Fill in the form:
   - **Title** (required) — A short description of the incident.
   - **Description** (optional) — A detailed description, supports Markdown.
   - **Escalation Policy** (required) — The escalation policy to trigger. The first available policy is pre-selected.
4. Click **Create**.

The incident is immediately created and the selected escalation policy is triggered. You can then acknowledge and resolve it like any other incident.



Viewing Incidents

The incident list shows all incidents with filtering by status. By default, active incidents (Unacknowledged + Acknowledged) are shown.

The sidebar shows a count badge on **Incidents** when there are active incidents (unacknowledged or acknowledged) open in your organization, and a separate badge on **My Incidents** for incidents you have acknowledged but not yet resolved. Badges refresh every minute; they are hidden when the count is zero. When the sidebar collapses to icon-only mode each badge shrinks to a small dot to signal that attention is needed.

The **My Incidents** sidebar entry is a shortcut that opens the incident list pre-filtered to incidents you have acknowledged — a quick way to see what you are actively working on.

- **Search** -- Find incidents by title or associated monitor name
- **Status Filter** -- Toggle visibility of each status type
- **Incident Type** -- Filter by how the incident was triggered: monitor failure, system incident, or manual incident
- **Monitor Type** -- Filter by the type of monitor that triggered the incident (Heartbeat, CVE, Trigger, Condition, Objective)
- **Tags** -- Filter by tags assigned to the monitor that triggered the incident
- **Escalation Policy** -- Filter by the escalation policy the incident is escalating under. This reflects the policy in use at the time the incident was raised, not any policy later reassigned to the monitor.
- **Acknowledged By** -- Filter by the team member who acknowledged the incident. Select **(me)** to show incidents you acknowledged.
- **Resolved By** -- Filter by the team member who resolved the incident. Select **(me)** to show incidents you resolved.
- **Time Range** -- Filter incidents by a time window. Click the clock icon to open the picker. Choose a quick preset or set a custom date and time range. Future dates cannot be selected.
- **Auto-Refresh** -- The list automatically refreshes every 5 seconds

On mobile, the search field is hidden by default — tap the search icon in the toolbar to reveal it. Tapping the icon again collapses the field and clears any active search.

On mobile, the filter controls are replaced by a **Filter** button that opens a bottom drawer. All filter groups (Status, Incident Type, Monitor Type, Tags, Escalation Policy, Acknowledged By, Resolved By) appear as expandable sections inside the drawer. Each section shows a count badge when filters are selected. Tap **Apply** to activate the selected filters or **Reset** to clear all draft selections. Active filters appear as read-only chips below the search bar; to change a selection, open the drawer again.

Each incident card in the list shows its status badge, monitor type, the tags assigned to the triggering monitor, and a brief cause. When the cause is identical to the incident title — common for simple monitor alerts — the cause line is omitted to reduce repetition.

Incident Details

Click on an incident to see its full details.

The detail page includes:

- **Status** -- Current incident state, shown as a colored text badge
- **Related Monitor** -- Link to the monitor that triggered the incident
- **Timestamps** -- When the incident started, was acknowledged, and resolved
- **Acknowledged By** -- Which user acknowledged the incident. Click the name to filter the incident list to all incidents they acknowledged.
- **Resolved By** -- Which user resolved the incident. Click the name to filter the incident list to all incidents they resolved.
- **Other Occurrences** -- Earlier incidents raised by the same condition (see below)
- **Activity Feed** -- Chronological log of all events

Other Occurrences

When the same monitor condition has raised incidents before, a card appears above the Activity Feed listing those earlier occurrences. Each row links to that incident and shows when it started and how long it lasted (or **still open** if it has not been resolved). The section only appears when at least one earlier occurrence exists.

Activity Events

The activity feed records events such as:

- Incident started
- Incident reopened (alert fired again within the reopen window configured on the escalation policy)
- Acknowledged by a specific user
- Resolved by a specific user
- Monitor failure details
- Monitor recovery details
- Email notifications sent
- SMS notifications sent
- Push notifications sent

- Voice call notifications sent
- Voice call ringing (attempted)
- Voice call declined
- Voice call not answered
- Alert email opened by a specific user
- Escalation paused — outside operating hours (shows the time escalation will resume)
- Sender timestamp changed (a later delivery for the same trigger alert carried a different source timestamp)
- Manually escalated to specific users, teams or on-call schedules (see below)

Source timestamp on trigger incidents — When a trigger monitor has a **Source timestamp** rule configured and the sending system includes a timestamp in the payload, the **Incident started** and **Incident reopened** events in the activity feed show the time the sender claimed the alert fired. Alongside it, the feed shows how far that time differs from when Pulse actually received the delivery — for example, "12 minutes before Pulse received it" or "3 minutes after Pulse received it — check the sender's clock". This is evidence about the sender's clock; it never changes when Pulse considers the incident to have started. See Trigger Monitor for how to configure source timestamp extraction.

Description on trigger incidents — When a trigger monitor's **Incident description** rule extracts text from the webhook payload, that text is shown on the incident detail page rendered as Markdown. A paging system's alert body often contains prose — links to the runbook, a list of affected assets, context that does not fit a title — and Markdown lets that formatting come through. When the description and the cause would say the same thing, the cause subtitle is hidden to avoid repeating it.

Escalating an Incident Manually

Sometimes the right people are not the ones the escalation policy would page next — an operator sees an alarm and knows the electrical maintenance team has to handle it, or has already acknowledged an incident and needs to pull in a specialist. **Escalate** sends a direct, immediate notification to people you choose.

To escalate an incident:

1. Open the incident detail page.
2. Click **Escalate...** (available until the incident is resolved — also on acknowledged incidents).
3. Pick one or more **users, teams or on-call schedules**. A team notifies all of its current licensed members; an on-call schedule notifies whoever it puts on call at that moment.

4. Tick the channels to alert them by: **Call**, **SMS**, **E-mail**, or **Push notification**. E-mail and push are pre-selected.
5. Click **Escalate**.

The chosen recipients are notified immediately, with a note naming who escalated to them, and the escalation is recorded on the activity feed together with the notifications that went out.

A few things to know:

- Manual escalation is **additive**: the automatic escalation policy keeps running on its own schedule until someone acknowledges the incident. Escalating never silences the policy.
- A manual escalation reaches the chosen people even outside their configured operating hours — it is a deliberate, named page.
- Recipients of a manual escalation also receive the follow-up notifications when the incident is later acknowledged and resolved.
- The activity feed names the on-call schedule you escalated to, not the people it resolved to — who was on call at that moment is visible in the notification entries that follow it.

Push Notifications

When an incident is created and an escalation step with push notifications enabled is triggered, recipients receive a push notification on their registered devices. When the incident is later **resolved**, those same recipients receive a follow-up push notification confirming the resolution.

Comments

You can add comments to an incident's activity feed to share updates, context, or next steps with your team.

To add a comment:

1. Open the incident detail page.
2. Click **Add Comment**.
3. Write your comment — Markdown is supported, including **task list checkboxes** (`- [ ] item`, `- [x] done`).
4. Click **Submit**.

The comment appears in the activity feed, visible to all team members.

Actions

From the incident detail page, you can:

- **Acknowledge** -- Mark the incident as seen and being worked on
- **Resolve** -- Close the incident as fixed
- **View Monitor** -- Navigate to the related monitor's detail page

If you resolve an incident while its monitor is still unhealthy, Pulse warns you before proceeding: resolving in this state immediately raises a new incident, and the comments and activity history of the current incident stay with it. The action button changes to **Resolve anyway** to confirm you want to proceed.

On mobile, the **Acknowledge** or **Resolve** action also appears as a full-width button pinned to the bottom of the screen for easy one-handed access. Tapping it opens a confirmation drawer before the action is carried out. If the monitor is still unhealthy, the confirmation drawer shows the same warning. The button disappears once the incident is resolved.

System Incidents

Pulse automatically raises system incidents when it detects operational issues that require administrator attention. These incidents appear in the same incident list as monitor-triggered incidents and trigger in-app notifications to all organization admins.

System incidents resolve automatically once the underlying condition clears — you do not need to resolve them manually.

Incident	Cause
Email delivery failure	An SMTP server failed to deliver a notification email.
No one available for escalation	An escalation policy was triggered but no recipient could be reached.
No one on call for [schedule]	A configured on-call schedule has no coverage during active operating hours.
Gateway disconnected	The appliance lost its connection to the Pulse gateway and could not reconnect.

Incident	Cause
Gateway enrollment failed	The appliance has not completed gateway enrollment. This appears during first-boot setup and resolves automatically once enrollment is finished.
No valid license	The organization has no active license. Some features may be limited.
License expiring soon	The active license is close to its expiry date.

3.5 Objectives

A monitor answers "is it broken right now". An objective answers a different question: **how much of the time has it been right, over a period you care about — and how much of your allowance for it being wrong is left.**

That second question is the one contracts are written in. "The line runs within limits 99.5 % of the month" is a promise a plant manager can hold you to; "coolant is currently 78 °C" is not. An objective is where you write the promise down, so Pulse can tell you whether it is being kept before somebody else does.

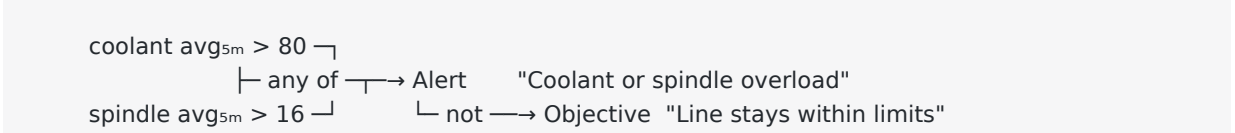
Pulse calls it an **objective**. Elsewhere you may see the same idea called an SLO — a service level objective. They are the same thing.

What an objective is about

An objective is a promise about a condition — specifically about one **Objective** node in that condition's graph.

That has a consequence worth understanding up front: **the condition defines what "good" means, and the objective defines how much good is enough.** They are edited in different places on purpose. Retargeting 99.5 % to 99.9 % is not a change to the logic, so it does not touch the graph, does not bump its version, and does not affect anything else reading that graph.

An Objective node is **true when things are as promised** — the opposite polarity to an Alert node, which is true when something is wrong. A graph carrying both usually looks like this:



One gate, two consumers, one `not` between them. Drawing it this way is what stops the page and the promise disagreeing about where the limits are: change the threshold once and both follow.

Several objectives may point at the same node. That is normal and useful — a contractual promise over a month and an operational one over a week are different promises about one indicator.

Creating one

From **Objectives → New objective**, which opens a dedicated create page. Fill in the promise fields and describe the indicator inline:

Field	What it means
Name	Names both the objective and the condition it is measured on
Target	How much of the *measured* time the condition must hold. Between 0 and 100 %
Window	How far back the promise looks, rolling. Between 1 hour and 365 days
Coverage floor	How much of the window must be measured before Pulse reports a figure at all
Escalation policy	Who is paged when the objective burns its budget — required, like any monitor
Tags	Optional labels, the same tags every monitor carries

The **indicator** section below the promise fields lets you describe what has to hold using a sentence form — a reading that stays within a range, or any threshold the appliance can evaluate. For anything a sentence cannot express, **Edit on the canvas** opens the full condition editor; save the drawn condition there and the editor offers **Create an objective**, which returns to this page with that condition already chosen. The section's **Use an existing condition** picker reaches the same state for a condition that is already stored — including one that already carries a promise, since one condition can back several objectives (say, 99.5 % over thirty days for the contract and 99.9 % over seven for the team). On both of those paths the condition is shown read-only and only the objective is created; the graph itself stays edited in the editor, where everything else consuming it is visible too. After the objective is created, its page links to the condition editor for edits.

The window is **rolling**, not calendar-aligned: a 30-day objective always means "the last 30 days", never "since the first of the month".

A **backtest** card appears once the target and coverage floor are filled in. It shows how the objective would have performed against the condition's history before you save — useful for checking that the window and target make sense on real data.

After creation, Pulse navigates directly to the new objective's page — and **backfills the window from stored sample history** in the background. Within moments the page shows a figure labelled *approximate* instead of sitting at "Not enough data" until enough live measurement has accrued. The backfilled stretch ages out of the rolling window as real measurement replaces it, and the label disappears with it.

Reading the figures

An objective's page leads with the achieved percentage and the coverage beside it. Underneath:

Figure	What it means
Error budget	How much out-of-spec time the target allows over this window
Spent	How much of it has been used
Budget left	What remains, as a percentage. Negative when you are past the allowance
Burn rate	How fast it is going. 1.0 is exactly on pace; 2.0 spends the whole window's budget in half a window
Unmeasured	Time in the window Pulse could not evaluate

Budget left is usually the number to act on. "32 % left" says what you can still afford; "99.66 %" says how it has gone so far.

An **approximate** badge next to the figure means part of the window is backfilled history — recomputed from stored samples when the objective was created, rather than measured live. The page says how much of the window that covers, and the burndown draws that stretch dashed. Backfilled time carries the same caveat a backtest does: it is noisy, not lenient. It counts for what the page shows, and never for burn alerts.

Coverage is always shown, and it is not decoration

Every compliance figure in Pulse carries the fraction of the window it was measured on, in the same breath. This is deliberate: **the standard way an availability figure lies is by quietly leaving out the time nobody was looking.** A perfect 100 % measured on two days of a thirty-day month is not a good month.

Time Pulse could not evaluate is excluded from the calculation rather than counted as good or as bad. Unknown is not failure, and it is not success either.

Below the coverage floor, there is no figure

If less of the window was measured than the coverage floor requires, the objective reports **Not enough data** instead of a percentage — the numbers behind it stay visible, but the headline is the caveat.

That is the honest answer, and it is worth preferring to a number: a promise measured on a third of its window is not being kept or broken, it is not being measured.

The budget burndown

The chart shows the error budget draining across the window — flat while the condition holds, stepping down whenever it did not. The exact date range covered is shown below the chart.

A burndown says something the percentage cannot: *when* the budget went. A single cliff on a Tuesday is one incident to investigate. A steady slope is a process that is not quite right and never triggers anything.

Being told about it

An objective **is a monitor underneath**: it carries an escalation policy, tags and a mute switch of its own, and its alerts raise incidents into the same list as everything else. Being paged about an error budget works exactly like being paged about a dead PLC — nothing new had to learn how to reach you.

What it pages about is decided by its **burn alerts** — the ladder in the panel of the same name on the objective's page. An objective with no rungs measures and reports and wakes nobody, which is a perfectly reasonable thing to have.

Burn alerts read **live measurement only** — backfilled approximate history never pages anybody. A freshly created objective therefore shows an approximate figure immediately but holds its alerts until enough real coverage exists, exactly as it would have without the backfill.

Why not just alert when the objective breaks?

Because by then it is too late to do anything about it.

A 30-day objective at 99.5 % has about three and a half hours of allowance. An alert that fires when the allowance runs out tells you about a month you have already lost. What you want to know is that you are **spending it too fast to last the month** — while there is still budget left to protect.

That is what a burn alert watches: not the level, the **rate**.

Setting up the ladder

In the **Burn alerts** panel on the objective's page:

1. Check who gets paged — the escalation policy was picked when the objective was created and can be changed here
2. Add one or more burn alerts — or take the recommended ladder
3. **Save**

The two kinds of burn alert

Kind	Reads
Budget dropping too fast	"At least X % of the budget went in the last W"
Time until the budget runs out	"At the current pace, the budget is empty within N"

Budget dropping too fast is the one to start with. It needs no extrapolation, and you can check it against the burndown chart on the same page — which is worth a great deal when deciding whether an alert that fired was right.

Time until the budget runs out is friendlier to write and noisier to run. It measures the current pace over a quarter of the horizon you ask about, so a short excursion can project to a breach that a longer look would not have predicted.

> **The percentage is of the whole budget, not of the window**
>
> "8.5 % within 1 hour" means eight and a half percent of the *whole month's* allowance, spent inside one hour.
>
> This is what makes the rungs comparable to each other and to the burndown. If each rung measured against its own window's share of budget, every threshold would mean something different and the ladder would be unreadable.

The recommended ladder

Two rungs, and the panel will fill them in for you:

Rung	Typically means
8.5 % in 1 hour	Something is badly wrong now — page somebody
43.2 % in 3 days	A slow bleed that will cost the month — raise a ticket

The shape matters more than the exact numbers: **one fast rung that catches a crisis, one slow rung that catches a drift.** A single rung either screams at every blip or misses the slow bleed entirely.

An objective has one escalation policy, so a ladder whose rungs must reach different people is drawn as two objectives over the same node — a fast promise routed to a policy that phones, a slow one to a policy that emails.

What an incident says

An incident is raised per rung, so two rungs firing are two incidents and clearing one does not clear the other.

> **Line 1 stays within limits is burning its error budget** 13.26 % of the error budget was spent within 1 h; 32.3 % of the budget remains.

It is named after the objective rather than after the rung, because that is the sentence that means something to whoever is reading it. Which rung it was is in the numbers. The incident links straight back to the objective's page.

Incidents escalate, are acknowledged, and are resolved exactly like any other. When the rung goes quiet, the incident resolves by itself.

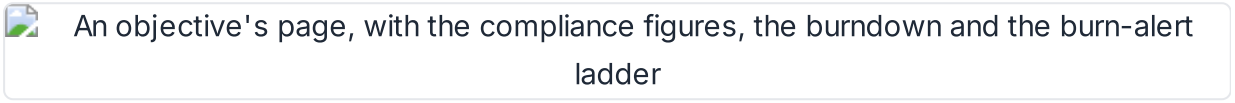
An unmeasurable objective holds its alerts open

If the objective drops below its coverage floor, its burn alerts read **Can't tell** rather than *quiet*.

An incident already open then stays open and **stops climbing the escalation policy** — Pulse will not wake more people over something it can no longer confirm, and it will not quietly declare all-clear either. Escalation resumes on its own when the data comes back.

This is the same treatment condition monitors get, and the reason is the same: a data point dying must never look like a problem being fixed.

Reading the ladder



An objective's page, with the compliance figures, the burndown and the burn-alert ladder

Each rung shows what it currently says, with the figure behind it:

Badge	Means
Burning · 13.26 %	Firing. That much of the budget went inside this rung's window

Badge	Means
13.26 %	Quiet, and this is how close it is
Can't tell	The objective is below its coverage floor — see above

Quiet rungs show their figure too, on purpose: the useful reading of a ladder is *how close* the quiet rungs are, not merely that they are quiet.

Editing the ladder

Rungs are edited together and saved together. A rung's window must be at least an hour and no longer than the objective's own window — a rung cannot ask about time the promise does not measure.

Removing a rung and saving retires it. Any incident it raised is resolved, and the rung's identity is not reused, so a later rung cannot inherit an old incident.

Muting

Mute in the objective's Actions menu (or in its list row's menu) silences it: burn alerts raise no incidents until it is unmuted. Measurement continues the whole time — the figures keep updating, and the page says plainly that the objective is muted.

It is the same mute every monitor has, because the objective is one.

Actions on the objectives list

Each objective in the list has a row menu with quick actions:

- **Open** — navigates to the objective's detail page.
- **Open condition** — opens the condition editor for the condition this objective is measured on.
- **Mute / Unmute** — the switch described above. A muted objective is marked in the list.
- **Delete** — removes the objective.

What happens to the condition when you delete an objective depends on whether anything else still uses it. If a monitor or another objective refers to the same condition, the condition and its history are left unchanged. If this objective is the only consumer, the condition is deleted with it. The delete dialog tells you which applies before you confirm.

Backtest

Nobody can pick a target they have never seen measured. The backtest recomputes what a promise *would* have said over a past range, from stored data point history, so you can choose a target with evidence instead of by guessing. It lives on the **create page**: once the objective exists, the same recomputation has already run automatically and its result is on the objective's page as the approximate backfill, so there is no second card to consult there.

It also answers **what spent the budget** — which branch of the condition was false while the budget drained, as a share of the spend. Because two branches can be false at the same moment, those shares do not add up to 100 %; each one reads "this part was false during N % of the spend".

> **A backtest is approximate, and the page says so**
>
> The recomputation samples on a coarse grid — the step is printed next to the figure. An excursion shorter than the step is either missed entirely or charged a whole bucket.
>
> That makes the result **noisy, not lenient**: it is as likely to overstate as understate, and the error shrinks with the number of excursions rather than with the length of the range. A month with three two-minute breaches can come back as zero, or as thirteen minutes.
>
> Use it to choose a target. Never quote it as the measurement — the recorded figure on the page above it is the measurement.

The backtest reaches back as far as data point history is retained, which is why the ranges offered are bounded.

In reports

Incident reports carry an objectives section: what each promise reported **over the report's period**, not over its own rolling window.

That substitution is the point. "Was this kept in June" is what a monthly report is asked, and answering it with a thirty-day rolling figure taken on the day the report was generated would silently include July and leave out most of June.

What Pulse cannot do here yet

Worth knowing before you promise something to somebody:

- **Planned downtime is not recognised.** Pulse has one kind of exclusion — absence of data — and it cannot tell a scheduled shutdown from a dead network. A plant that stops every weekend will sit below its coverage floor and report *Not enough data* rather than a number. **If your plant does not run continuously, read this before setting a coverage floor.**
- **Burn alerts are evaluated once a minute**, not on the five-second monitor loop. The shortest legal rung window is an hour, so a minute is well inside the resolution the question has.
- **Windows are rolling only.** A contractual "calendar month" is not yet expressible.
- **One condition per objective.** Promises about several things at once are drawn as one condition with an `all of` gate, not as several objectives.
- **The condition can change underneath a window.** If it does, the page says *The condition changed during this window* — the figure then covers two different definitions of "good", which is not wrong but is worth knowing before quoting it.

3.6 Reports

Incident Reports let operators schedule recurring, shareable summaries of their incidents. Each report covers a complete period — a day, week, or month — and is delivered automatically to the people who need to see it.

Report Specs

Reports are created and managed on the **Reports** page (`/app/reports`). Creating, editing, and deleting report specs is admin-only. The list shows each spec's cadence, next scheduled time, and recipient count. Specs that include external email recipients are marked with an **External** label.

A report spec defines:

- **Name** — A short label for the report.
- **Cadence** — **Daily**, **Weekly**, or **Monthly**. The report covers the last complete period for that cadence, evaluated in the organization's timezone.
- **Send hour** — The hour of day (0–23, organization timezone) at which the report is generated and sent.
- **Link validity (days)** — How long a shared report link stays valid.
- **Filters** — Restrict which incidents appear in the report. You can filter by incident **statuses**, **monitor types**, **data sources**, and **tags**. Each filter is optional; leaving one empty applies no restriction on that dimension. The **data sources** filter limits the report to incidents from monitors that read the selected data sources — useful for a device-specific report. For the **tags** filter: within a single tag category, any matching tag is sufficient (OR logic); across categories, a monitor must carry at least one matching tag in each selected category (AND logic).
- **Recipients** — Two kinds. **Email recipients** are external people who do not have a Pulse login; their copy is published to a password-protected public URL emailed to them directly. **Pulse users and teams** are members of your organization; picking a team sends the report to everyone in it. Team membership is resolved each time the report is generated, so people who join a team later are included automatically.
- **Enabled** — When off, the report is saved but will not generate automatically on its cadence. You can still generate it manually.

Generation

Reports generate automatically on their cadence at the configured send hour. Admins can also open the **Actions** menu on a spec and choose **Generate now** to produce a report immediately. The same **Actions** menu contains the **Delete** option to remove a spec.

Report History

Every generated report is **frozen** — it captures the incident data at the moment it ran and never changes afterward, even as the underlying incidents are acknowledged, resolved, or edited.

A spec's past reports are listed on its detail page under **Recent reports**, each showing its period, when it was generated, and when it expires. The list shows the most recent reports first; use **Older** and **Newer** to page through the history.

Reports are kept until they expire. Once a report passes the expiry date shown next to it — its spec's **Link validity (days)** counted from when it was generated — it is removed automatically, both from this list and from the shared link its recipients hold. If you need a report to stay available longer, raise **Link validity (days)** on the spec before the reports you care about are generated.

What a Report Contains

Each report summarizes the incidents in its period across several sections:

- **Headline counts** — Incidents opened, resolved, and still open, plus MTTA and MTTR.
- **Volume & trends** — How incident volume changed over the period.
- **Status at end of period** — How many incidents were Unacknowledged, Acknowledged, or Resolved when the period closed.
- **When incidents start** — Onset heatmap broken down by weekday and hour.
- **Where incidents cluster** — Incidents grouped by monitor type, by tag, and by top monitors.
- **Responder load** — How the response workload was distributed across your team.
- **Reliability** — Objectives first: for each objective, what it promised (target), what it achieved, how much of the period was actually measured, and how much of the error budget was spent. An objective measured on too little of the period reports **Not enough data** rather than a figure. Under each objective sits its **error budget**

burndown, showing how the budget was spent across the period — the line starts full and only falls, so its slope is the burn rate and the zero line is the point the promise was used up. Then recurring monitors and system events over the period.

- **Incident detail** — A filterable, sortable, exportable table of individual incidents.

Viewing a Report

How a report is opened depends on who is viewing it.

Pulse users — whether picked individually or through a team — open reports in-app, at an authenticated view that needs no password. The report is reachable from the spec's report list, and is linked directly from the in-app notification and email they receive.

External email recipients open reports through a password-protected shared link. Each recipient receives their own distinct random password, emailed to them once — it is never stored. Opening the link prompts for that password and then shows the report. The public report viewer includes a **language toggle** (EN / DE) so recipients can switch the report's display language. Links expire after the spec's **Link validity (days)**.

Delivery

When a report generates:

1. **Email recipients** are emailed their personal shared link together with their password.
2. **Pulse users**, including the members of every selected team, receive an in-app notification and an email that links to the authenticated in-app view. Someone who is both picked individually and part of a selected team is notified once, not twice.

See Incidents for details on the incidents that feed these reports.

3.7 Vulnerabilities

Pulse matches published CVEs (Common Vulnerabilities and Exposures) against your device inventory and surfaces each match as a **finding** you can triage. Unlike an incident — which represents a monitor going up or down — a vulnerability rarely resolves on its own. It needs a decision: does it apply to this device, is the device used in a way that exposes it, and how will you handle it? Vulnerabilities are therefore a first-class area in Pulse with their own lifecycle, activity log, and reporting, separate from incidents.

CVE monitors

A **CVE monitor** represents one device in your inventory that you want scanned for known vulnerabilities. You create it with the device's identity:

- **Vendor** — the device manufacturer (for example, Siemens).
- **Order code** — the manufacturer's order number (the Siemens MLFB, for example `6ES7515-2AN03-0AB0`).
- **Firmware** (optional) — the installed firmware version, so version-specific advisories can be evaluated precisely.

CVE monitors live alongside your other monitors under **CVE Monitors**. You can create one directly, or start from a device's data source, which pre-fills the vendor and links the monitor to that source. The link is fixed when the monitor is created — to point it at a different source, create a new monitor.

Pulse re-scans each CVE monitor regularly against the vulnerability catalog. A monitor is shown as **unhealthy** while it has any vulnerability still awaiting action, and healthy once every finding has been triaged away or no vulnerabilities apply.

Findings and their lifecycle

Each applicable CVE becomes a **finding** attached to its CVE monitor. A finding carries the advisory details (CVSS score and severity, affected product and version, remediation guidance, and a link to the source advisory) plus a **triage status**:

- **Open** — newly detected, not yet triaged.
- **In progress** — actively being investigated or remediated.
- **Resolved** — the vulnerability has been remediated.

- **Won't fix** — a real vulnerability, but a deliberate decision not to remediate (accepted risk). Requires a reason.
- **Not affected** — triaged as not applicable (for example, the affected feature isn't used, or the version match was a false positive). Requires a reason.
- **Risk accepted** — temporarily accepted until a date you choose. When that date passes, the finding automatically returns to **Open** for re-triage. Requires a reason and a future expiry date.

Triage decisions are **sticky**: they persist across re-scans. If a vulnerability disappears from a scan and later reappears, its previous triage decision is kept rather than reset.

Findings are never deleted. When a re-scan no longer detects a CVE, the finding is marked **no longer detected** and retained, so your history and trends stay complete. Use the **Show no-longer-detected** filter to include these in the list.

Triaging a finding

Open a finding to see its full advisory, triage panel, and affected-devices list.

The **Affected devices** panel lists every device in your organization that has a finding for the same CVE — not just the one you are looking at. Each entry shows the device name, its triage status, and whether the CVE is still detected. Click any device to jump to its own finding. The current device is labeled *(this device)* and is not a link. Use this panel to understand the blast radius of a CVE across your fleet and to triage consistently.

Administrators can change the triage status from the **Change triage** action:

1. Choose the new **status**.
2. For **Won't fix**, **Not affected**, or **Risk accepted**, enter a **reason**.
3. For **Risk accepted**, pick the date the acceptance expires.
4. Save.

Every member can add **comments** to a finding to record investigation notes.

Activity feed

Each finding keeps a chronological **activity feed** of everything that happened to it — automatic scan events (detected, no longer detected, reappeared, risk-acceptance expired) and human actions (triage changes with the before/after status and reason, and comments). Scan-driven entries are attributed to *System*; human entries show who made them.

The **Activity** tab on the Vulnerabilities page shows the same events across every finding in your organization, newest first, so you can see recent triage and detection activity at a glance.

The Vulnerabilities page

The **Vulnerabilities** page has four tabs:

- **Overview** — a summary dashboard with counts by triage status and severity, a 30-day detected-vs-resolved trend, and the most affected devices.
- **CVEs** — one expandable row per unique CVE, aggregated across all your devices. Each row shows the CVE id, title, how many devices are affected, the severity, and a breakdown of triage statuses across all findings of that CVE. Click a row to expand it and see the individual findings inline.
- **Findings** — one row per individual finding (a flat list).
- **Activity** — the org-wide activity feed (see Activity feed).

Filtering and searching

The **CVEs** and **Findings** tabs share the same filter and search controls:

- **Search** — by CVE id, title, or affected product.
- **Filter** — by triage status, severity, and monitor tags; and toggle whether no-longer-detected findings are shown.

Bulk triage

From the **CVEs** view, organization administrators can triage all findings of a CVE at once without opening each one individually. Click the ⋯ menu on a CVE row and choose **Triage all**. The dialog works the same as the single-finding triage dialog — choose a status, add a reason if required, and confirm. The decision is applied to every finding of that CVE across your organization's devices.

If filters are active when you open the bulk triage dialog, a notice reminds you that the action applies org-wide, not only to the filtered subset visible in the current list.

Bulk triage events appear in the activity feed with the label **Bulk-triaged**, so you can distinguish them from single-finding triage changes.

Investigating with AI Canvas

On any CVE finding detail page, the **Ask AI** button opens the AI Canvas with a pre-filled question about that specific CVE and device. The assistant explains the vulnerability in plain terms — what it is, how severe it is, and what an attacker could do — and then analyzes how it affects the particular monitored device: whether it is likely exploitable in an OT environment and what concrete remediation or mitigation steps to take.

You can also ask the AI Canvas about vulnerabilities without opening a specific finding. From the canvas, the assistant can query your CVE findings by triage status, severity, affected device, and other filters — for example, "which of my devices have critical open CVEs?" or "give me a vulnerability report for my Siemens devices."

Reporting

Pulse emails a weekly **vulnerability summary** to organization administrators: counts by triage status plus the number of new critical and high-severity findings detected in the past week. No configuration is required.

Licensing

The vulnerability feature is enabled per organization by your license. Without it, CVE monitors and findings are unavailable.

Part 3 · Monitoring

3.8 AI Canvas

AI Canvas is an interactive investigation workspace where you ask questions about your system and the assistant lays out its findings on a spatial canvas. Each question starts a new canvas; the assistant's reasoning — incidents queried, monitors checked, time-series data retrieved — appears as cards you can pan and zoom around.

> **INFO**
>
> AI Canvas requires an active license with the AI Canvas feature enabled. If your license does not include AI Canvas, the **Canvas** entry does not appear in the sidebar. Contact your administrator to upgrade your license.

Getting started

Navigate to **Canvas** in the sidebar to open the canvas list. Type a question in the prompt and press **Ask** (or **Enter**) to start a new canvas. Pulse sends the question to the AI assistant and opens the canvas as the assistant begins working.

You can also start a canvas from the **Query** page: click the **Ask AI for help** button (✦) in the query builder toolbar to open a new canvas with a pre-filled prompt that guides the assistant to help you build a time-series query.

You can ask about anything related to your monitored system — for example:

- "Are there any active incidents on the production line?"
- "Show me the signal trend for Sine0 over the last hour."
- "Which monitors are currently unhealthy?"
- "Which of my devices have critical open CVEs?"

The canvas

Once the assistant has returned results, the canvas opens in a split view with two separate panels:

- **Left — investigation board**: The assistant's findings appear as cards on a free-form spatial canvas. Each card represents one tool result — a data query, incident list, monitor list, or CVE findings. Cards appear as the assistant pins them and stay on the board between sessions.

- **Right — conversation**: The chat panel shows the full conversation: your questions and the assistant's text responses. Tool calls the assistant made are accessible here via a **Show tool calls** toggle.

When the board is empty — before the assistant has pinned anything — it shows a **Nothing pinned yet** message as a reminder that findings land here and the conversation stays in the chat.

The assistant presents times in your organization's configured timezone when writing responses and reports. Internally it works in UTC; the conversion is automatic.

Cards

Each card on the board shows the result of one tool call the assistant made:

Card type	What it shows
Data query	A time-series chart from a data point query
Incidents	Incidents matching the assistant's search
Monitors	Monitors matching the assistant's search
CVE findings	Vulnerability findings matching the assistant's search
Tags	Tags matching the assistant's search

Each card shows a timestamp indicating when it was captured. Data query cards have three tabs — **Chart**, **Overview**, and **Values** — for different views of the same result. When a tool returned more rows than it can display, the card shows a notice at the bottom.

Incidents, monitors, and CVE findings in cards link directly to their detail pages. When a result set is long, a **Show more** toggle lets you expand the full list.

You can resize any card using the width control in its header: **Narrow**, **Wide**, or **Extra wide**.

Navigating the board

Action	How
Pan	Drag empty board space, or hold **Space** and drag
Zoom	Ctrl / ⌘ + scroll wheel, or pinch on a trackpad
Scroll (pan without zoom)	Scroll wheel (without Ctrl / ⌘)

Action	How
Zoom in / out	**+** / **−** buttons in the toolbar
Fit all cards to view	⛶ (fit) button in the toolbar
Reset zoom to 100%	**Reset zoom** in the toolbar

The current zoom level is shown as a percentage between the zoom buttons.

Board toolbar

The toolbar at the bottom of the board has two interaction modes:

- **Select** — Click cards to select them; drag cards to reposition them on the board.
- **Pan** — Drag to pan the board without selecting or moving cards. Useful when many cards are close together.

Switch between modes using the **Select** and **Pan** buttons in the toolbar. Holding **Space** temporarily switches to pan mode regardless of which mode is active, so you can pan without switching modes.

Tidy the board rearranges all cards into a clean grid, which is useful after a long session where cards have been scattered. The positions after a tidy are saved like any other drag.

Arranging cards

Cards can be freely dragged to any position on the board. Their positions are saved automatically and shared with anyone viewing the same canvas.

On touch devices and small screens the board is read-only — cards cannot be dragged or resized. The full arrangement is still visible; editing requires a mouse and a larger screen.

Renaming a canvas

Canvas owners can rename a canvas by clicking its title in the header bar. Edit the name and press **Enter** to save, or **Escape** to cancel.

Pulse also names the canvas automatically in two stages: as soon as you send the first message, the canvas takes a name from the first line of your prompt so it is identifiable while the run is in progress. After the run completes successfully, the assistant replaces that interim name with a short, polished title derived from the full conversation. A manual rename at any point locks the name — neither automatic stage overwrites it.

A canvas that has not yet received an automatic title appears as **Untitled canvas** in the list, the switcher, and the title bar. Clicking the title to rename starts with an empty field so you do not have to clear the placeholder text first.

The auto-generated title is written in the same language as your conversation — asking questions in German produces a German title.

Switching canvases

The canvas editor header contains a **Switch canvas** button (stack icon) on the right side. Click it to open a side panel listing all of your canvases. Select any row to jump directly to that canvas, or click **New canvas** at the bottom to return to the canvas list and start a fresh one.

Board and Chat tabs (small screens)

On small screens where the board and chat cannot appear side by side, two toggle buttons — **Canvas** and **Chat** — appear below the header bar. Use them to switch between the investigation board and the conversation panel. On larger screens both panels are always visible and the toggle is hidden.

Continuing the investigation

Owners can send follow-up questions using the text field at the bottom of the chat panel. Press **Enter** to send (or **Shift+Enter** to insert a new line). The assistant runs again and adds new cards to the canvas.

Non-owners see a read-only notice at the bottom of the chat panel that names the canvas owner and offers a link to start their own canvas. If the owner has left the organization, the notice says so instead.

Managing canvases

The canvas list page shows all canvases you own or can view. Each row shows the canvas name, owner, and when it was last updated. A colored status badge appears next to the timestamp for canvases that are running (pulsing amber) or failed on their last run (red).

When your list grows beyond a handful of canvases, a **Filter** input appears above the list. Type any part of a canvas name to narrow the list. The filter applies only to the loaded list — it does not search the server.

To delete a canvas, open the row's action menu (⋯) and choose **Delete**. Confirm the deletion in the dialog. Deletion is permanent — the canvas history and all its findings are removed.

Related documentation

- Query — Build ad-hoc time-series queries on your monitor data directly.
- Monitors — The monitors the assistant can query on your behalf.
- Incidents — The incidents the assistant can search and summarize.
- Vulnerabilities — CVE findings the assistant can query and analyze.

3.9 Query

The Query page lets you explore the time-series data Pulse collects from your data sources — OPC UA servers, S7 PLCs, Prometheus endpoints, and heartbeats. Instead of looking at a single monitor, you build an ad-hoc query — pick the data points, choose how to aggregate them, and see the result as charts or a table.

This is read-only analysis. Queries never change your monitors, thresholds, or incidents.

How a query is built

A query is assembled in the **Define** panel from three parts, shown side by side:

- **SELECT** — the aggregations to compute (e.g. average, maximum, a percentile).
- **WHERE** — the data points to include.
- **GROUP BY** — how to split the results into separate series.

The time range and granularity are set from the clock icon in the panel header. Press **Run Query** to execute, or **Cancel** to clear the current results.

> **Ask AI for help**
>
> Click the **Ask AI for help** button (✦) in the toolbar to open AI Canvas with a prompt pre-filled to guide the assistant through building a query. The assistant asks what you want to find out, constructs the query, and shows you the results.

SELECT — aggregations

Each aggregation reduces all the readings in a time bucket to a single number. You can add up to **four** aggregations to one query.

An aggregation is a function applied to a field:

- **Functions** — Count, Sum, Average, Min, Max, Latest, and percentiles (P1, P5, P10, P25, P50/median, P75, P90, P95, P99, and finer steps such as P0.1 and P99.9).
- **Fields** — the value being aggregated:
  - **Value** — the numeric reading from the data point. Boolean readings count as `0` or `1`.
  - **Latency** — how long the read took, in milliseconds. Useful for spotting slow or degrading connections.

`Count` does not need a field — it counts the number of samples in each bucket. `Latest` returns the value at the sample with the greatest timestamp in the bucket — useful when you want the most recent reading rather than an aggregate like the average or maximum.

To add an aggregation, start typing a function name. Type the function followed by `(` to choose a field — for example `avg(` then pick **Value** or **Latency**. The result reads back as a chip like `avg(value)` or `max(latency)`.

> **TIP**
>
> Only successful reads are included. A reading is counted only when the source reported a good status (OPC UA status *Good*, S7 read status *OK*). Failed or stale reads are excluded, so the aggregates reflect healthy data only.

WHERE — data points and tags

The WHERE field selects which data points feed the query.

Start typing a data source name to find it, then pick one of its data points — an OPC UA node, an S7 address, a Prometheus series, or a heartbeat. Choose `*` to include everything that source measures at once. Selected data points appear as chips; remove one with its × or by pressing Backspace.

You can mix data points from several sources, of any kind. They are queried together and combined into one result set.

> **What `*` leaves out**
>
> `*` covers what a source measures, not its health signals. A source's own reachability, and any OPC UA node watched for whether it still exists rather than for its value, are left out — each reports a constant, and averaging one in beside a temperature would flatten the result. Select either by name to chart it on its own.

You can also filter by **monitor tags** — the labels you assign to monitors. Tags appear in the same dropdown, organized by category. Select one or more tags to restrict the query to data from monitors that carry those tags. Tag chips show the tag color and name alongside any selected data point chips. Remove a tag chip with its × or by pressing Backspace. A query may use tags alone, without selecting specific data points.

> **WARNING**
>
> At least one data point or tag is required. A query with an empty WHERE field will not run.

GROUP BY — splitting into series

By default, every selected data point is aggregated together into a single series. Group By splits that single series into several:

- **Monitor** — one series per condition monitor that reads the selected data points.
- **Data Point** — one series per data point.
- **Data Source** — one series per data source.

You can combine them. `avg(value)` grouped by **Data Point** gives one line per point, which is the quickest way to compare readings against each other. Add **Monitor** to break each point out by the monitor watching it — a point read by two monitors then appears once under each.

Time range and granularity

Open the **clock** icon in the panel header to set:

- **Time Range** — a quick range (last 15 minutes through last 30 days) or a custom start/end.
- **Granularity** — the size of each time bucket (10 seconds up to 1 week). Leave it on **Auto** to let Pulse pick a sensible bucket for the chosen range. When granularity is not set explicitly, Pulse buckets data by the hour.

Finer granularity gives more detail but more points; coarser granularity smooths the data and runs faster over long ranges.

Reading the results

After a query runs, the charts appear below the builder, with an **Overview** and a **Table** view in a tabbed control beneath them. A small summary line shows the query's execution time, the number of time buckets returned, and the granularity that was actually applied.

Charts

The charts are always shown. Each aggregation in your SELECT gets its **own chart**: a query with `count` and `avg(value)` produces two separate graphs — one per aggregation — so different units and scales never share a single axis. Within each chart, every Group By series is drawn as its own line.

The colors are explained by the Overview table below, where each row carries the same color as its line.

Incident markers

Incidents that occurred on the queried monitors during the time window are drawn on the charts as vertical markers, colored by status (unacknowledged, acknowledged, resolved). Use the **Incidents** button above the charts to show or hide individual incidents — handy for lining up a spike in the data with the incident it caused.

Overview

The **Overview** tab summarizes the whole window as a table with **one row per group**. Each aggregation is a column (e.g. `count`, `avg(value)`), and when you group by Monitor, Data Point, or Data Source those appear as columns too. A colored dot on each row matches that group's line in the charts.

These totals are computed over **all** readings in the window — so `avg(value)` is the true average across the window, percentiles are exact, and `count` is the total — not a combination of the per-bucket numbers.

Table

The **Table** tab shows the time-series breakdown — one row per time bucket and group, with a column for each aggregation. Columns are sortable. Use the table when you want the exact per-bucket numbers rather than a trend or a window total.

Suggested queries

Below the builder, **Suggested Queries** offers ready-made starting points for common questions, such as average value per data point, peak values, polling latency, the slowest data sources, and sample volume per data source.

Selecting a suggestion fills in the SELECT, GROUP BY, granularity, and time range for you while keeping the data points you have already chosen. If you have not selected a data point yet, pick one in the WHERE field first, then run the suggestion.

The suggestions disappear once you run a query and return when you press **Cancel**.

On mobile, the panel is replaced by a **Suggested Queries** button. Tapping it opens a bottom drawer that lists all suggestions with their names and descriptions. Tap any suggestion to run it immediately.

Related documentation

- Monitors — the condition monitors that alert on the readings you query here.
- OPC UA, Siemens S7, and Heartbeat — how the underlying data is collected.

- AI Canvas — ask the assistant to build and run queries on your behalf.

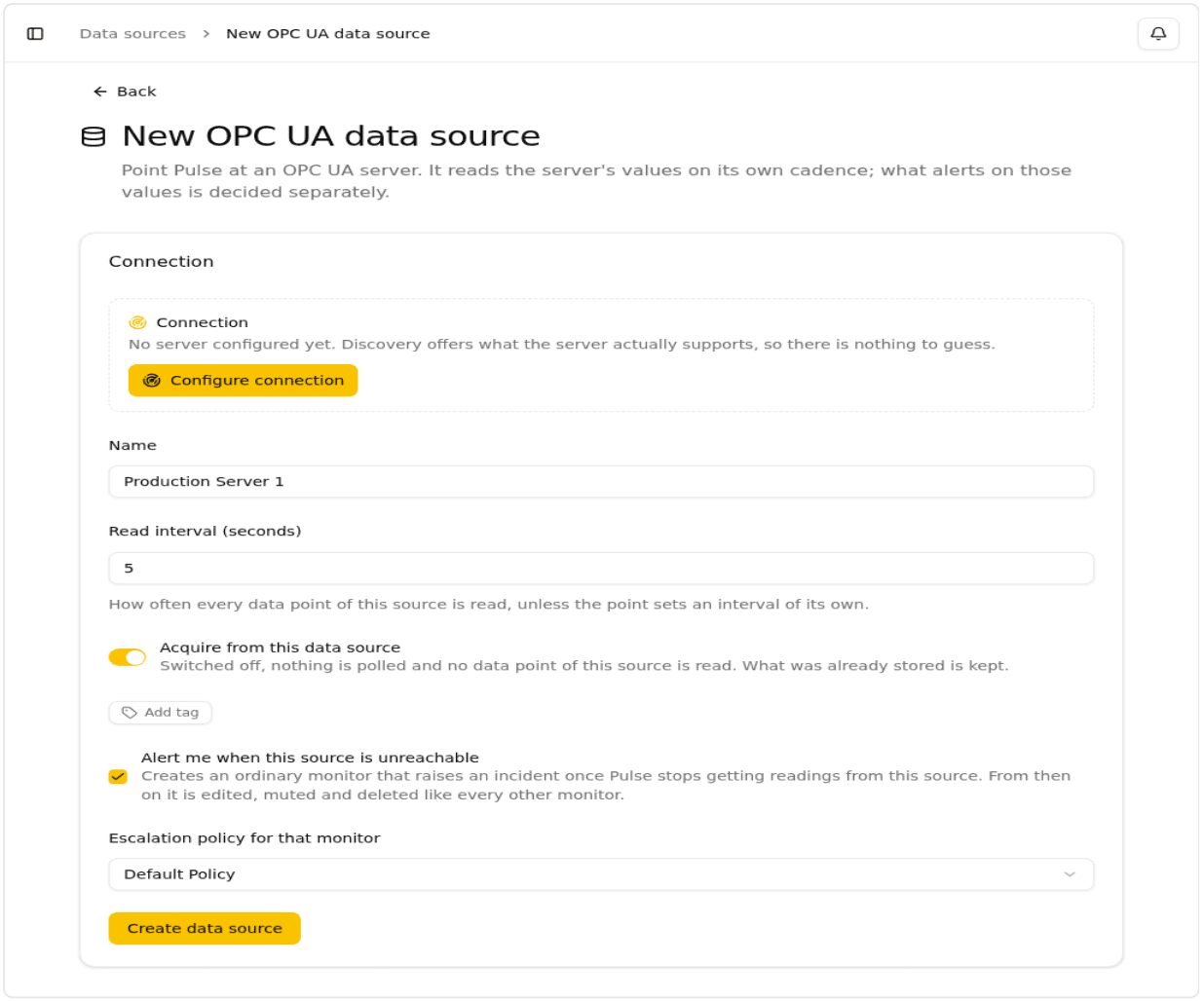
Part 4 · Monitor Types

4.1 OPC UA

OPC UA (Open Platform Communications Unified Architecture) is an industrial communication protocol for secure, reliable data exchange. Pulse connects to an OPC UA server as a **data source** and reads its values.

Adding an OPC UA Data Source

A data source holds everything about one server connection in one place: its address, how the channel is secured, how Pulse authenticates, and how often values are read.



1. Navigate to **Data sources → OPC UA** in the sidebar, which lists every OPC UA server Pulse reads, and click **New data source**
2. Select **OPC UA**
3. Configure the source:

Field	Description	Example
Name	A descriptive name for the source	`Production Server 1`
Connection	Endpoint URL, security mode and policy, authentication, and the server's pinned certificate — all configured together	`opc.tcp://192.168.1.100:4840`
Read interval (seconds)	How often every data point of this source is read	`10`
Acquire from this data source	Switch this off to stop polling without deleting the source	
Tags	Categorize the source; every data point inherits its tags	
Alert me when this source is unreachable	Creates a monitor alongside the source (see below)	

Click **Configure connection** to open the connection panel, where the endpoint URL, security mode, security policy, authentication, and the pinned server certificate are set as one unit. The form then shows a summary of what you committed; **Edit** reopens the panel.

Security settings

The security mode, security policy, authentication, endpoint discovery, and both certificates involved in the connection are explained in depth in OPC UA Security. In short: use **Sign & Encrypt** with **Basic256Sha256** or stronger in production, and let **Discover endpoints** pick the strongest configuration your server offers.

NETWORK REQUIREMENTS

Ensure your firewall allows TCP connections on the OPC UA port (default: 4840) between the Pulse server and your OPC UA servers.

Being Alerted When the Server Goes Down

Alert me when this source is unreachable is ticked when you add a source. It creates a condition monitor named after the source — *Production Server 1 unreachable* — that raises an incident once Pulse stops getting readings from it. Pick the escalation policy it should use in the field below the tick box.

From the moment it exists, it is an ordinary monitor: rename it, mute it, change its escalation policy, edit its condition, or delete it, all from the monitor list. Untick the box if nobody should be paged for this particular server.

Adding Data Points

The individual values on a server are **data points** of the source. The source's detail page lists every one Pulse reads, with its node ID, last value, last sample time, and reading quality.

To declare a new data point, open the source's detail page and choose **Add data point** from the actions menu. The dialog has two tabs:

- **Browse** — connects to the OPC UA server and shows its node tree. Expand nodes to find the values you want to read, tick one or more, and click **Add**. Nodes already declared on this source are marked and skipped.
- **Type manually** — enter the node's OPC UA identifier directly, e.g. `ns=2;s=Temperature` or `ns=2;i=1001`.

The data point's value type is filled in once the first reading arrives. Pulse starts polling the new node on the source's next cycle.

Alerting on a Value

To alert on a data point, build a condition and attach a condition monitor to it. A condition compares readings against thresholds, aggregates them over a time window, and can combine several of them — including a reading and its source's reachability in one rule, so an unplugged cable raises the one incident that says the cable is unplugged instead of one per value read through it.

The source's detail page shows a **Monitors** section listing every condition monitor that alerts on any data point of this source. Click a monitor to open its detail page. A source with no monitors alerts on nothing; that section says so explicitly — a useful reminder that readings are being collected but nobody is paged when they go wrong.

Each row in the data points table also shows which monitors alert on that specific node. A single monitor appears as an icon button that opens it directly; when two or more monitors alert on the same node, the button shows the count and opens a picker listing them by name.

Data points build up a history only while you keep them. Use **Keep** on the source's detail page for the values you want to chart and query over the long term; the rest stay readable for 24 hours.

OPC UA Alarms & Conditions

OPC UA Alarms & Conditions (A&C) is a standard mechanism for servers to push event and alarm notifications to clients. When an OPC UA data source receives an A&C event notification, Pulse automatically creates an incident for it.

Each incident created from an A&C event:

- Uses the alarm message from the OPC UA server as the incident title
- Displays the same alarm message in the incident detail view as the description, so the full alarm text is always visible in notifications and incident details

Incidents from A&C events are deduplicated: if the same condition triggers multiple times, only one open incident is created per condition.

Application instance certificate

To open a secure channel (**Sign** or **Sign & Encrypt**), Pulse presents an **application instance certificate** — a self-signed X.509 identity, one per organization, shared by every OPC UA data source. Each OPC UA server must **trust** this certificate before it will accept a secure connection.

Managing the certificate — downloading it, trusting it on your servers, and revoking it if compromised — is covered in OPC UA Security — Application instance certificate.

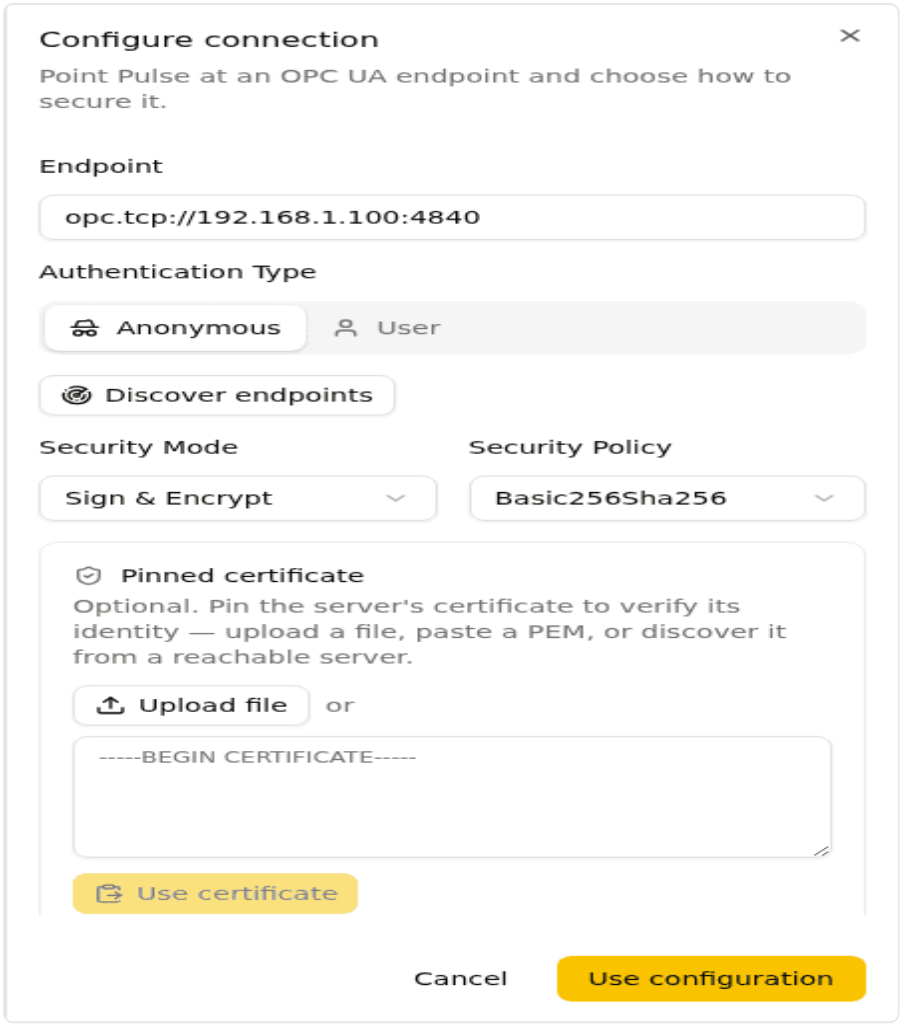
Rotation — replacing an expiring certificate

Because a replacement certificate is a new identity, Pulse rotates with an **overlap window**: a replacement is minted ahead of time, you trust it on every server while monitoring keeps running on the old one, then you complete the rotation. The full workflow is described in OPC UA Security — Rotation.

Part 4 · Monitor Types

4.2 OPC UA Security

Every OPC UA data source decides how its connection to the server is secured. This page explains the pieces of that decision: the **security mode** and **security policy** that protect messages in transit, the **two certificates** involved — Pulse's own identity and the server's — and how the session is **authenticated**.



How a secure connection works

An OPC UA connection is secured in two layers:

1. **Secure channel** — the transport layer. Pulse and the server exchange X.509 certificates and, depending on the security mode, sign and/or encrypt every message. Pulse presents its application instance certificate; the server presents its own certificate, which Pulse can pin.

2. **Session** — inside the channel, Pulse authenticates as a user: anonymously or with a username and password. See Authentication.

Two practical consequences follow:

- For **Sign** or **Sign & Encrypt**, the server must **trust Pulse's certificate** before any connection succeeds.
- Encryption alone does not verify **who** you are talking to. Only pinning the server's certificate does.

Security mode

The security mode sets what the secure channel does with each message:

Mode	Effect	Use when
None	Messages are sent in plain text; no certificates are exchanged.	Only in isolated networks where the path is trusted.
Sign	Messages are signed — tampering is detected, but content is readable.	Integrity matters, confidentiality does not.
Sign & Encrypt	Messages are signed and encrypted.	Recommended for production.

Mode and policy are coupled: selecting mode **None** forces the policy to **None** (and vice versa), and switching to a secure mode preselects **Basic256Sha256**.

Security policy

The security policy is the algorithm suite used for signing and encryption:

Policy	Status	Notes
None	—	No cryptography. Only valid with security mode None.
Basic128Rsa15	Deprecated	Legacy SHA-1-based suite. Only for old servers that offer nothing else.
Basic256	Deprecated	Legacy SHA-1-based suite. Only for old servers that offer nothing else.

Policy	Status	Notes
Basic256Sha256	Current standard	Recommended default — widely supported.
Aes128Sha256RsaOaep	Current	Modern alternative to Basic256Sha256.
Aes256Sha256RsaPss	Current	Strongest suite — use it if the server offers it.

> **TIP**
>
> Pick the strongest policy your server supports. Discovery does this for you: it lists only combinations the server actually offers and preselects the strongest one.

Finding the right configuration with discovery

Instead of guessing what the server supports, let Pulse ask it. With an `opc.tcp://` endpoint URL filled in, click **Discover endpoints** in the connection panel. Pulse queries the server and shows:

- **Security configuration** — the real mode + policy combinations the server offers, strongest first. The strongest is marked **Recommended**; combinations without signing or encryption carry a warning.
- **Server certificate** — subject, issuer, validity dates, and thumbprint of the certificate the server presented, with badges for expired or soon-expiring certificates. For a secure configuration, this certificate is offered as the pin.

If the server is unreachable, you can still choose the security mode and policy manually — discovery is a convenience, not a requirement.

The two certificates

Secure OPC UA connections involve one certificate per side, and each side must decide to trust the other:

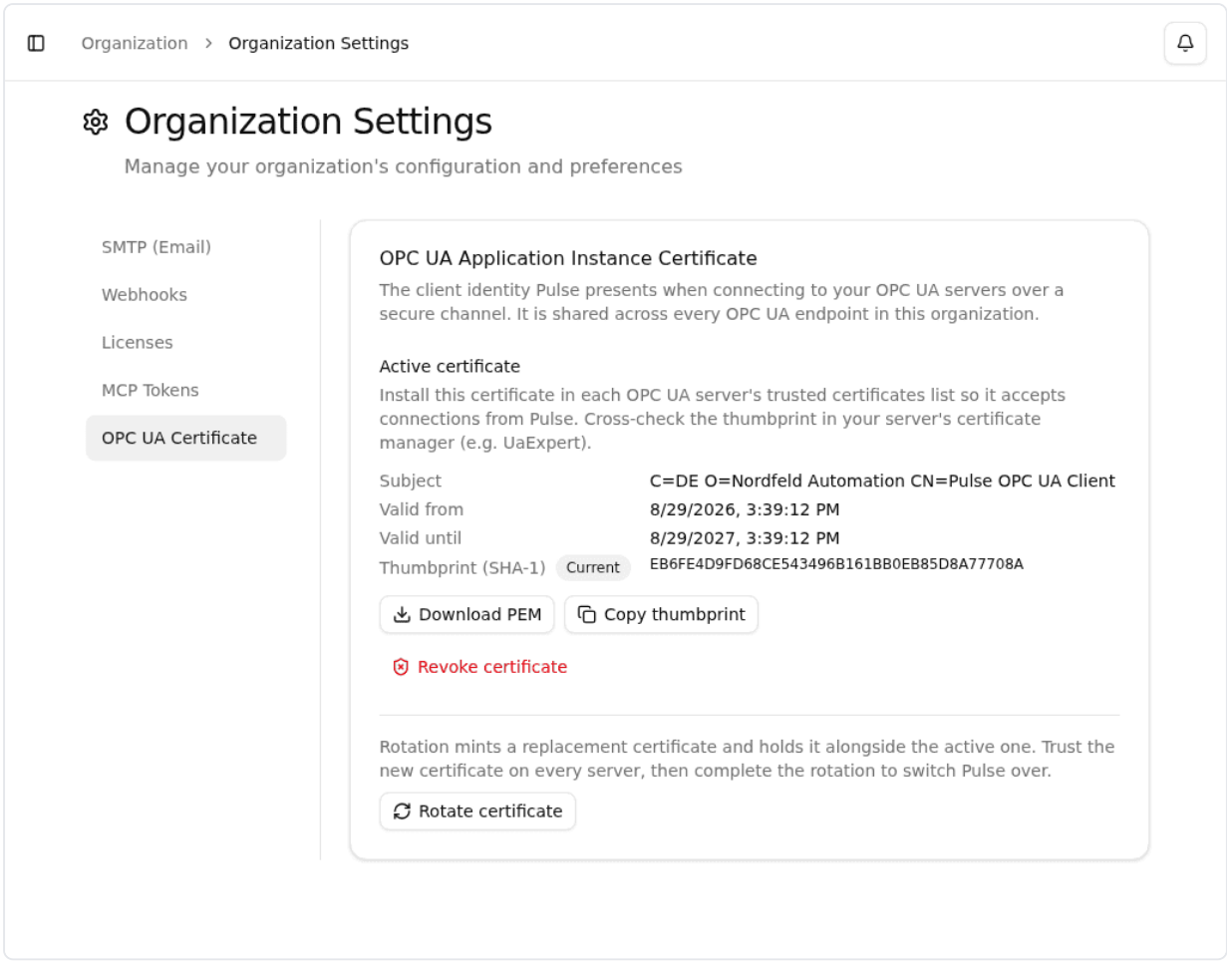
Certificate	Who presents it	Who must trust it
Application instance certificate	Pulse	Your OPC UA servers — otherwise they reject the connection.

Certificate	Who presents it	Who must trust it
Server certificate	The server	Pulse — verified only if you pin it to the data source.

Application instance certificate

To open a secure channel (**Sign** or **Sign & Encrypt**), Pulse presents an **application instance certificate** — a self-signed X.509 identity, one per organization, shared by every OPC UA data source. Each OPC UA server must **trust** this certificate before it will accept a secure connection.

You manage the certificate under **Organization → Settings → OPC UA Certificate**. Pulse provisions it automatically the first time a secure connection needs it, or you can click **Generate certificate** ahead of time so you can trust it on your servers before adding data sources.



Property	Value
Subject	`Pulse OPC UA Client` (self-signed)
Key	RSA 2048-bit, SHA-256 signature
Validity	One year; rotation replacements ~14 months, so a full year remains after the switch
Thumbprint	SHA-1 of the certificate, uppercase hex

From the settings page you can **Download PEM** and **Copy thumbprint** to cross-check against what your server shows in its trust list.

Trusting the certificate on your servers

The exact steps depend on your OPC UA server, but the pattern is always the same:

1. Download the certificate from **Organization → Settings → OPC UA Certificate**.
2. Import it into the server's **trusted certificates** store (in UaExpert and many servers this is the "Trusted" or "PKI" folder).
3. Confirm the thumbprint the server shows matches the one in Pulse.

Until the certificate is trusted, secure-channel handshakes to that server fail.

Rotation — replacing an expiring certificate

Because a replacement certificate is a **new identity**, every server that trusted the old one must trust the new one — so Pulse rotates with an **overlap window** rather than a hard swap:

1. **A replacement is prepared ahead of time.** About 60 days before expiry (or immediately, if you click **Rotate certificate**), Pulse mints a **pending** certificate and keeps serving the current one. The settings page then shows both, Pulse raises an incident, and your organization's admins receive an email.
2. **You trust the new certificate everywhere.** Copy the pending certificate's thumbprint and add it to the trusted store on every OPC UA server — exactly as you did for the current one. Monitoring keeps running on the old certificate throughout.
3. **You complete the rotation.** Once every server trusts the new certificate, click **Complete rotation**. Pulse promotes the pending certificate to active and removes the old one. New secure channels use the new identity immediately.

> **WARNING**
>
> Only complete the rotation **after** every server trusts the new certificate. Completing it early breaks secure connections to any server that has not yet trusted it.

Started a rotation you don't want? **Discard replacement** deletes the pending certificate and leaves the active one untouched.

If a certificate expires before you complete the rotation, Pulse promotes the waiting replacement automatically — the old certificate is already unusable at that point, so this can only restore monitoring, never break it. You still need to have trusted the new certificate on your servers for connections to recover.

If the certificate is compromised

Revoke certificate immediately discards the active certificate and generates a new identity. Every OPC UA server will reject Pulse until you trust the new certificate on it, so secure monitoring stays down in the meantime. Use it only for a compromised key — for a planned replacement, rotation avoids the outage.

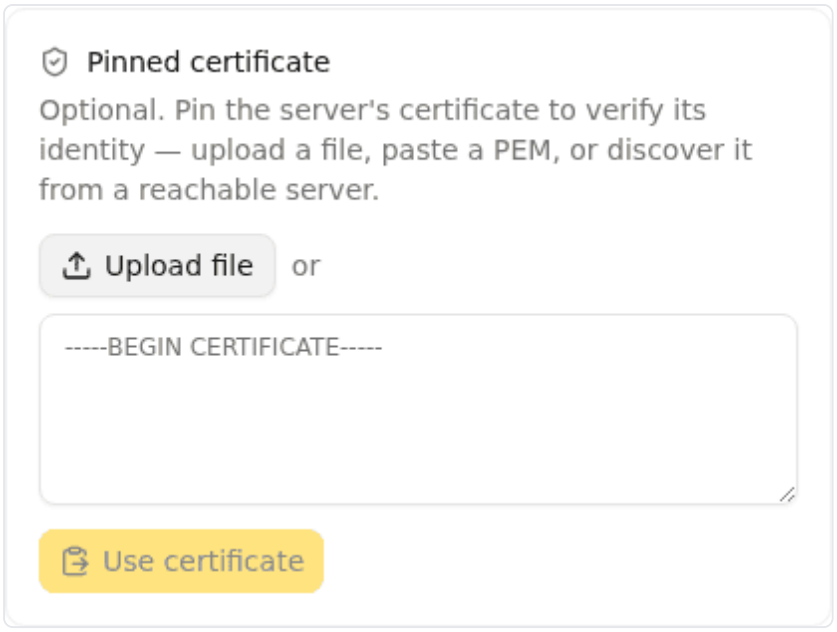
Server certificate pinning

Pinning is how Pulse verifies the **server's** identity — the other half of the trust relationship.

Without a pin, Pulse accepts whatever certificate the server presents. The channel is still encrypted under Sign & Encrypt, which protects against eavesdropping — but not against impersonation: anything that answers on that address could terminate the connection. With a pin, future connections must present **exactly** the pinned certificate, or the connection fails.

Pinning is optional and available whenever the security mode is Sign or Sign & Encrypt. There are three ways to pin:

- **Discovery** — run discovery and the server's current certificate is offered as the pin. Recommended when the server is reachable.
- **Upload file** — upload a PEM or DER certificate file.
- **Paste PEM** — paste the certificate text and click **Use certificate**.



Once pinned, the section shows the certificate's thumbprint and validity. **Replace** supplies a different certificate; **Unpin** removes it (after a confirmation, since it removes the man-in-the-middle protection).

When the server's certificate changes

If you run discovery on a data source that already has a pin, Pulse compares the two and tells you whether the pinned certificate **matches** or **differs from** what the server currently presents. A difference can mean a legitimate certificate rotation on the server — or a man-in-the-middle. Adopting the new certificate is therefore always an explicit action: verify the change is expected, then click **Use discovered certificate**.

> **WARNING**
>
> If you change the endpoint URL while a certificate is pinned, the pin belongs to the previous server and is removed when you save — a warning in the form makes sure this is never silent. The same applies when switching the security mode to None. Re-pin after the change.

Pinning and stored credentials

If a data source uses **username/password authentication** on a signed or encrypted channel **without** a pinned certificate, the form shows a warning: the server's identity is unverified, so a man-in-the-middle could intercept the credentials. Pinning the server certificate closes this gap — treat the warning as a prompt to pin, not as noise.

Authentication

Independent of the transport security, the server may require Pulse to authenticate as a user:

- **Anonymous** — no credentials; the server must allow anonymous sessions.
- **User** — username and password. Credentials are stored in Pulse as named, organization-wide entries and selected per data source; organization admins can create new ones directly from the connection panel.

> **TIP**
>
> When using username/password authentication, run the connection with **Sign & Encrypt** and a pinned server certificate so credentials can neither be read in transit nor handed to an impostor.

Network requirements

Pulse connects over the binary `opc.tcp://` protocol — the endpoint URL must use this scheme. Ensure your firewall allows TCP from the Pulse server to the OPC UA server's port (conventionally **4840**, but servers can listen anywhere — use the port from your endpoint URL).

Troubleshooting

Symptom	Likely cause	Fix
Secure connection fails right after enabling Sign / Sign & Encrypt	The server does not trust Pulse's certificate yet	Trust the application instance certificate on the server
Connection fails after maintenance on a source with a pinned certificate	The server's certificate was rotated	Open the connection panel, run discovery, verify the new certificate, then adopt it
Secure connections fail after completing a certificate rotation	A server was missed when trusting the new certificate	Trust the new thumbprint on that server

Symptom	Likely cause	Fix
Authentication errors	Wrong username/password, or the server account lacks rights	Check the stored credentials and the account's permissions on the server
The chosen security configuration is rejected	The server no longer offers that mode + policy combination	Run discovery again and pick a configuration the server offers
Server unreachable	Network path or firewall	Verify the endpoint URL, port, and firewall rules

Related

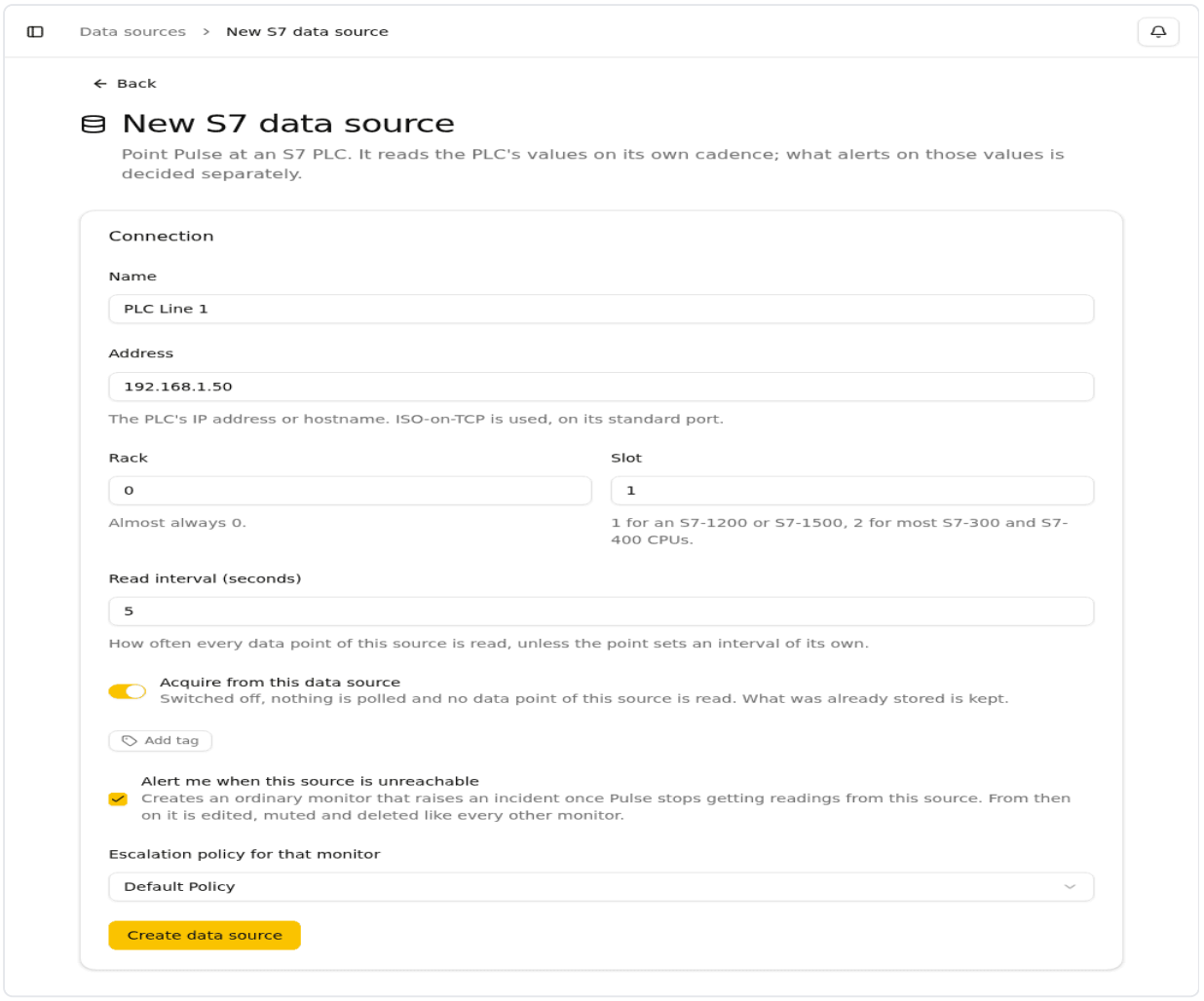
- OPC UA Configuration — adding an OPC UA data source
- Monitors — monitor concepts shared by all types

Part 4 · Monitor Types

4.3 Siemens S7

Pulse reads Siemens SIMATIC S7 PLCs using the S7 communication protocol. A PLC is added as a **data source**: one connection that carries every value Pulse reads from that machine.

Adding an S7 Data Source



1. Navigate to **Data sources → S7** in the sidebar, which lists every PLC Pulse reads, and click **New data source**
2. Select **S7**
3. Configure the connection:

Field	Description	Typical Value
Name	A descriptive name for the source	`PLC Line 1`
Address	The PLC's IP address or hostname	`192.168.1.50`
Rack	PLC rack number	`0`
Slot	PLC slot number	`1`
Read interval (seconds)	How often every data point of this source is read	`10`
Acquire from this data source	Switch this off to stop polling without deleting the source	
Tags	Categorize the source; every data point inherits its tags	
Alert me when this source is unreachable	Creates a monitor alongside the source (see below)	

Pulse connects over ISO-on-TCP on its standard port, so no port needs to be configured.

Rack and Slot Values

Common configurations for Siemens PLCs:

PLC Type	Rack	Slot
S7-1200	0	1
S7-1500	0	1
S7-300	0	2
S7-400	0	2 or 3

Being Alerted When the PLC Goes Down

Alert me when this source is unreachable is ticked when you add a source. It creates a condition monitor named after the source — *PLC Line 1 unreachable* — that raises an incident once Pulse stops getting readings from it. Pick the escalation policy it should use in the field below the tick box.

From the moment it exists, it is an ordinary monitor: rename it, mute it, change its escalation policy, edit its condition, or delete it, all from the monitor list. Untick the box if nobody should be paged for this particular PLC.

Adding Data Points

The individual values in PLC memory are **data points** of the source. The source's detail page lists every one Pulse reads, with its S7 address, last value, last sample time, and reading quality.

To declare a new data point, open the source's detail page and choose **Add data point** from the actions menu. The dialog asks for:

- **Address** — the S7 address in the same syntax the sampler reads, e.g. `I0.0`, `MW10`, or `DB1.DBD4`.
- **Data type** — how Pulse interprets the raw PLC memory at that address (see Data Types below).

The address and data type must agree: a `DBW` (Word) address holds a 16-bit value, so it pairs with **Word** or **Int**, not with **Real**. Pulse starts reading the new data point on the source's next poll.

To declare many data points at once from a spreadsheet, use **Import** from the same actions menu. See Importing from CSV for the full column reference, including the optional threshold columns that create a condition monitor alongside each data point.

Alerting on a Tag Value

To alert on a data point, build a condition and attach a condition monitor to it. A condition compares readings against thresholds, aggregates them over a time window, and can combine several of them — including a reading and its source's reachability in one rule, so a PLC that drops off the network raises the one incident that says so instead of one per value read from it.

The source's detail page shows a **Monitors** section listing every condition monitor that alerts on any data point of this source. Click a monitor to open its detail page. A source with no monitors alerts on nothing; that section says so explicitly — a useful reminder that readings are being collected but nobody is paged when they go wrong.

Each row in the data points table also shows which monitors alert on that specific point. A single monitor appears as an icon button that opens it directly; when two or more monitors alert on the same point, the button shows the count and opens a picker listing them by name.

Data points build up a history only while you keep them. Use **Keep** on the source's detail page for the values you want to chart and query over the long term; the rest stay readable for 24 hours.

S7 Address Format

S7 addresses follow the pattern `DBx.DByz` where:

- `DB` -- Data Block prefix
- `x` -- Data Block number
- `y` -- Data type: `B` (Byte), `W` (Word/2 bytes), `D` (DWord/4 bytes)
- `z` -- Byte offset within the data block

Examples:

Address	Description
`DB1.DBW0`	Word at offset 0 in Data Block 1
`DB2.DBD4`	Double word at offset 4 in Data Block 2
`DB10.DBB0`	Byte at offset 0 in Data Block 10

Data Types

Each S7 data point carries a data type that tells Pulse how to interpret the raw PLC memory at its address:

Data Type	Description
Bit	Single boolean value (0/1)
Byte	Unsigned 8-bit integer
Word	Unsigned 16-bit integer
DWord	Unsigned 32-bit integer
Int	Signed 16-bit integer
DInt	Signed 32-bit integer
Real	32-bit floating point number

The address and the data type must agree: a `DBW` (Word) address holds a 16-bit value, so it pairs with **Word** or **Int**, not with **Real**.

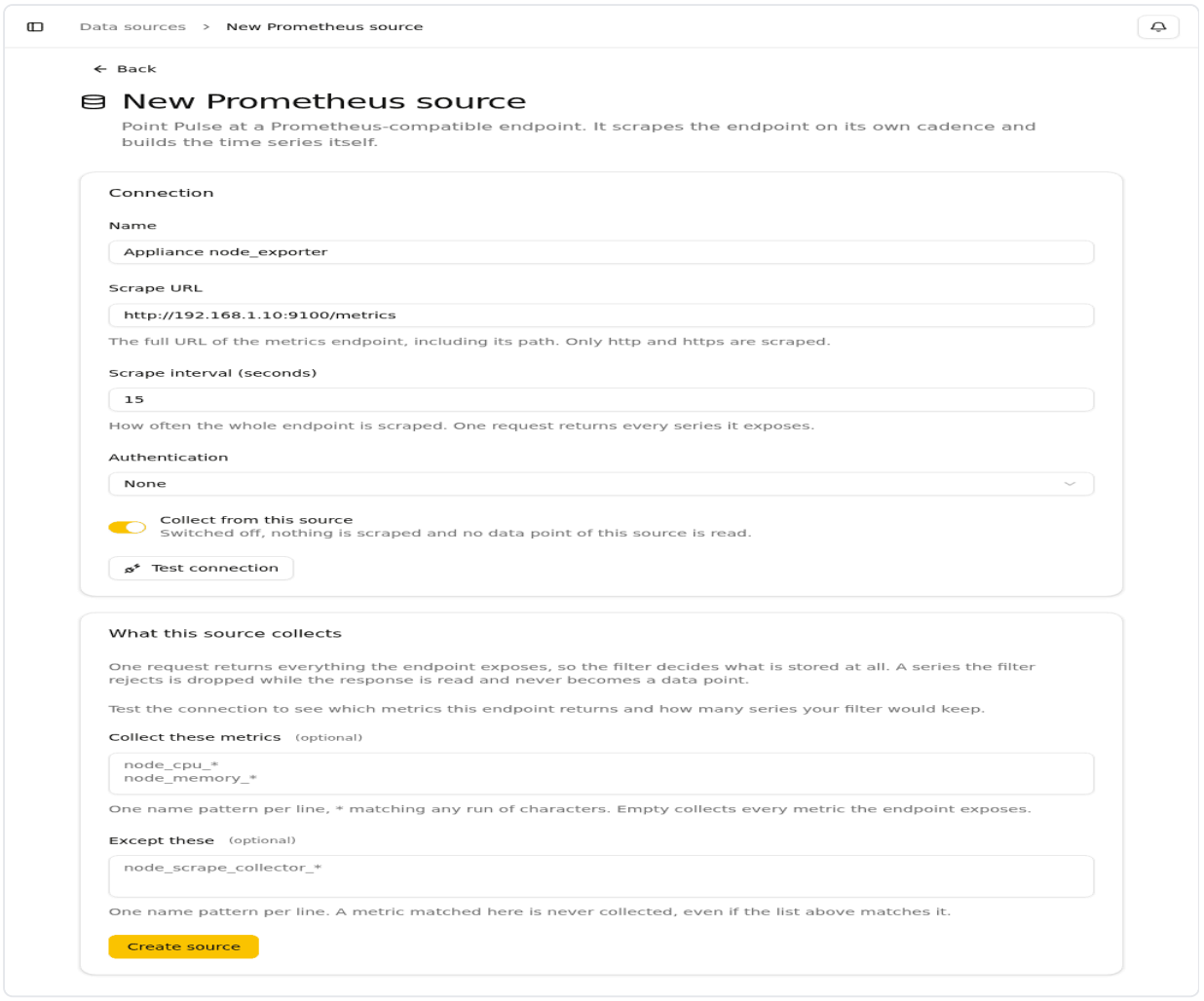
> **TIP**
>
> Refer to your PLC program documentation for the correct data block numbers, offsets, and data types for the values you want to read.

Part 4 · Monitor Types

4.4 Prometheus

Pulse connects to a Prometheus-compatible metrics endpoint as a **data source**. It scrapes the endpoint on its own cadence, stores the returned time series as data points, and makes them available for conditions and charting.

Adding a Prometheus Data Source



1. Navigate to **Data sources → Prometheus** in the sidebar and click **New data source**
2. Select **Prometheus**
3. Configure the source:

Field	Description	Example
Name	A descriptive name for the source	`Appliance node_exporter`
Scrape URL	The full URL of the metrics endpoint, including its path	`http://192.168.1.10:9100/metrics`
Scrape interval (seconds)	How often the endpoint is scraped; one request returns every series it exposes	`15`
Collect from this source	Switch this off to stop scraping without deleting the source	
Authentication	How Pulse authenticates to the endpoint (see below)	

Authentication

Three modes are available:

Mode	When to use
None	The endpoint is open or protected by network controls
Bearer token	The endpoint requires an `Authorization: Bearer <token>` header
Basic auth	The endpoint requires a username and password

Secrets (tokens and passwords) are stored write-only: Pulse never displays them again after saving. Leave the field blank when editing to keep the stored secret; enter a new value to replace it.

Testing the connection

Click **Test connection** before saving. Pulse scrapes the endpoint once and reports how many series across how many metrics it found. This step is important: one scrape request may return hundreds of series, and you want to see what you are keeping before it starts accumulating.

What this source collects

Every Prometheus scrape returns every series the endpoint exposes. The **metric filter** decides which of those series are stored as data points; series the filter rejects are dropped while the response is read and never become data points or consume storage.

The filter has two lists:

List	Behaviour
Collect these metrics	One name pattern per line. `*` matches any run of characters. Leave empty to collect every metric the endpoint exposes.
Except these	One name pattern per line. A metric matched here is never collected, even if the collect list above matches it.

After you run **Test connection**, the filter panel shows a live count of how many series your current filter would keep out of the total returned. Adjust the patterns and watch the count move before you save.

> **Start broad, then narrow**
>
> Leave the filter empty on first save, browse what the source collects on its detail page, and then edit the source to add a deny list for the series you do not need. This is easier than writing patterns blind against metrics you have never seen.

Data points and history

Once a source is collecting, each distinct time series becomes a **data point** visible on the source's detail page. The page shows:

- The metric name and any labels
- The last value read and when it was read
- Whether the reading quality is Good, Bad, or No data

Data points build up a history only while you keep them. Use **Keep** on the source's detail page for the series you want to chart or query over the long term; the rest remain readable for 24 hours and are then dropped.

Alerting on a value

To alert when a Prometheus metric crosses a threshold, build a condition using a **Sensor** node pointed at the data point. A condition can combine several data points in one rule — for example, alert when CPU usage climbs above the learned normal level.

Metric types

Pulse stores Prometheus metric types as they arrive and makes them available for querying:

Type	Description
Counter	A monotonically increasing counter; use a **Rate of increase** node in a condition to convert it to a per-second rate
Gauge	A value that can go up or down
Histogram	A sample distribution with count, sum, and bucket series
Summary	A pre-computed quantile distribution
Untyped	Metrics the endpoint did not type

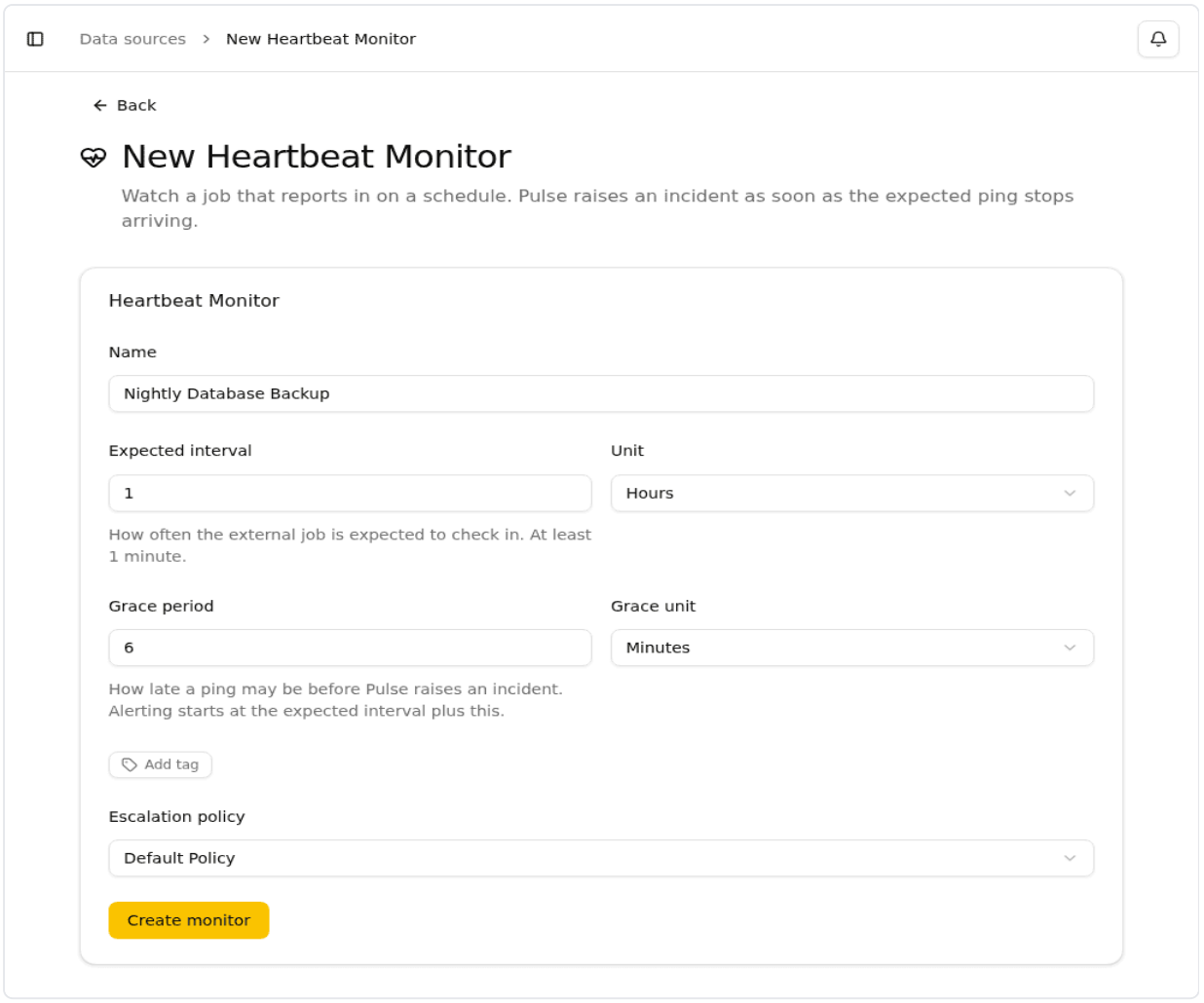
4.5 Heartbeat

A heartbeat monitor watches a job that runs on a schedule -- a nightly backup, an hourly export, an edge device that reports in. The job calls a URL every time it finishes successfully, and Pulse raises an incident when that call stops arriving.

This is the one monitor type that works the other way round: Pulse does not reach out to anything. It waits, and silence is the alarm. That makes it the right choice for anything Pulse cannot reach directly -- a cron job on a machine behind a firewall, a script on a laptop, a gateway on a remote site.

A heartbeat is a **data source**: Pulse acquires one signal from it -- "a ping arrived" -- and an ordinary monitor alerts when that signal goes quiet. You will find your heartbeats under **Data sources**, and the monitor that pages about each one in the monitor list beside everything else.

Creating a Heartbeat



1. Navigate to **Data sources** and click **New**, or use **Create** on the monitor list
2. Select **Heartbeat**
3. Fill in the fields:

Field	Description
Name	A descriptive name, usually the job's name (e.g. `Nightly Backup Job`)
Expected interval	How often the external job is expected to check in. Minimum: 1 minute
Unit	Minutes, Hours, or Days
Grace period	How late a ping may be before Pulse raises an incident. Alerting starts at the expected interval plus this
Grace unit	Seconds, Minutes, or Hours

Field	Description
Escalation policy	Which escalation policy to use when the ping stops
Tags	Categorize the heartbeat

Creating a heartbeat creates its monitor at the same time, named after it. That monitor is an ordinary one: you can mute it, hand it to a team, change its escalation policy, or open it and add a condition of your own alongside the ping check.

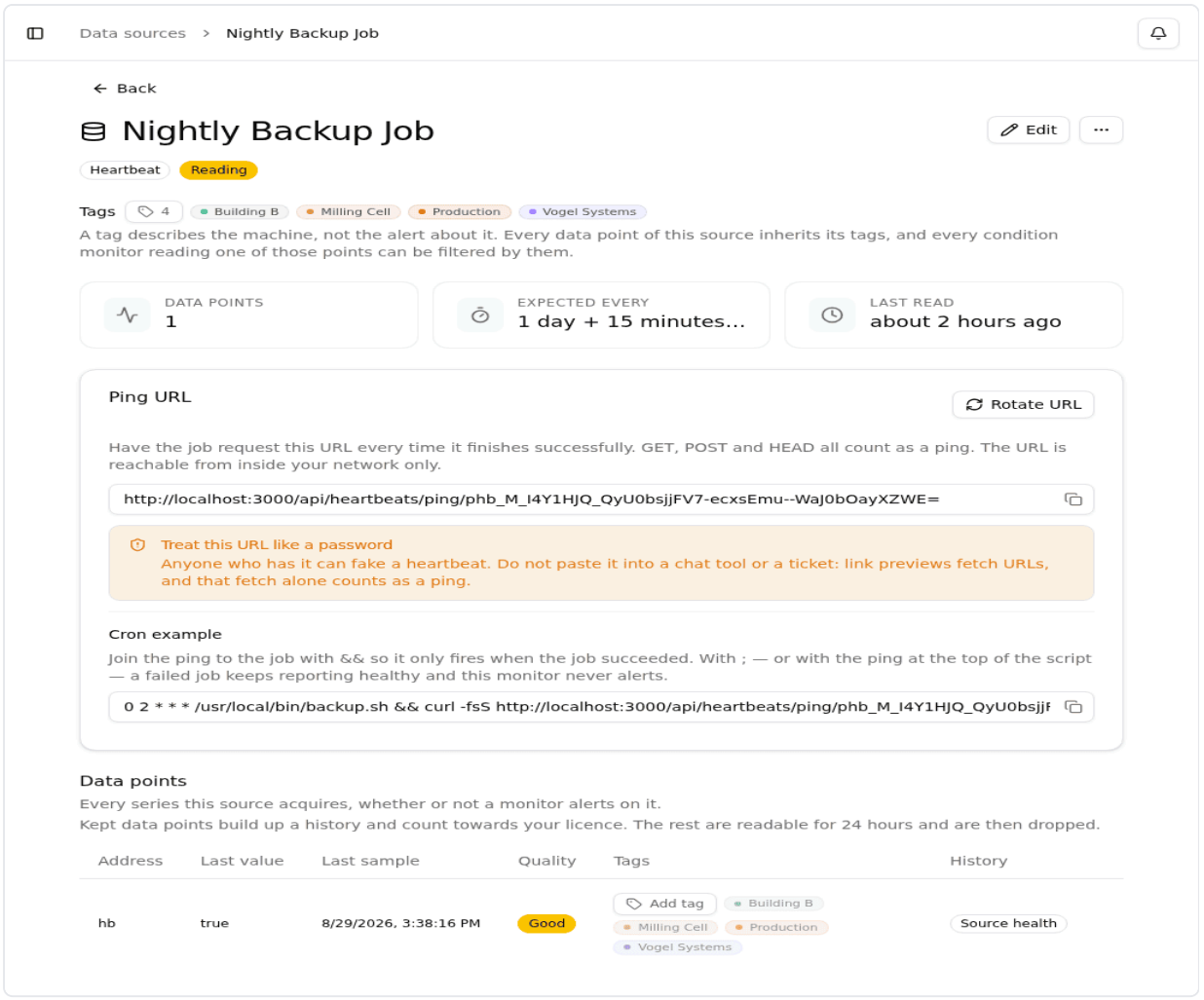
The **expected interval** is the schedule your job runs on. Set it to match how often the job actually runs — not longer.

The **grace period** covers the jitter in when a job finishes. A backup that runs every hour but occasionally takes a few extra minutes should have a grace period of those few minutes, so a slow run does not page anyone. Pulse raises an incident only after both the expected interval and the grace period have elapsed without a ping. A grace period of zero means Pulse alerts the instant the expected interval passes.

The clock starts the moment you create the monitor, not at the first ping. If no ping ever arrives, the monitor raises its first incident after the expected interval plus the grace period have elapsed. Until either the first ping or that first deadline, the monitor's status is **Unknown**.

The Ping URL

Each heartbeat gets its own ping URL, shown on the heartbeat's page under **Data sources**.



Click the copy button next to the URL to put it on your clipboard. Any of `GET`, `POST`, or `HEAD` counts as a ping, so whatever your job already has available -- `curl`, `wget`, a scheduled task, a two-line script -- will work.

The URL is reachable from inside your network only. A job running on a machine that cannot reach Pulse cannot ping it.

Treat the ping URL like a password

Anyone who has the URL can fake a heartbeat, and a faked heartbeat means a failed job that reports healthy.

Do not paste it into a chat message, a ticket, a wiki page, or a shared document. Chat tools and ticket systems generate link previews by **fetching the URL** -- that fetch alone counts as a ping, so pasting the URL somewhere convenient can silently keep the monitor green for as long as the preview cache refreshes.

Put the URL where secrets already live: your configuration management, a secrets manager, or directly in the job's own script on the machine that runs it.

Rotating the Ping URL

If the URL leaked, click **Rotate URL** on the heartbeat's page under **Data sources**. The old URL stops working immediately, so install the new one right away -- a job still calling the old URL no longer counts as a ping, and the monitor raises an incident once the expected interval has passed.

Rotating only replaces the URL. The monitor's history, its expected interval, and any open incident are left alone.

Wiring It Into Cron

Put the ping **after** the job and join the two with `&&`:

```
0 2 * * * /usr/local/bin/backup.sh && curl -fsS https://pulse.example.com/api/heartbeats/ping/your-tol
```

`&&` means "only if the previous command succeeded". If the backup fails, the ping never runs, Pulse hears nothing, and you get the incident you wanted.

Three ways to get this wrong, all of which leave a monitor that reports healthy forever while the job is broken:

- **Joining with `;` instead of `&&`.** A semicolon runs the ping whether the job succeeded or not.
- **Putting the ping first.** `curl ... && /usr/local/bin/backup.sh` pings before the job has done anything at all.
- **Putting the ping at the top of the script.** Same problem, one file further away and much harder to notice in review.

The rule is simple: the ping is the job's success signal, so nothing may ping unless the job has finished and succeeded.

The `-fsS` flags on `curl` keep it quiet on success and make it fail on an HTTP error, so a network or server problem shows up in your cron mail instead of disappearing.

Note that a *mistyped or rotated-out* ping URL will not show up there: the ping endpoint always answers `200 OK`, whether or not the token is valid, so that nobody can use it to test whether a token they found is live. A wrong URL surfaces the same way a job that stopped running does -- the monitor stops hearing from you and raises an incident once the expected interval passes. After you set up or rotate a URL, check the monitor's **Last ping** to confirm it arrived.

For a job that is a longer script rather than a single command, put the ping on the last line and make sure the script exits early on failure:

```
#!/bin/sh
set -e
run_the_backup
verify_the_backup
curl -fsS https://pulse.example.com/api/heartbeats/ping/your-token
```

With `set -e`, any failing step aborts the script before it reaches the ping.

When the Ping Stops

Once the expected interval plus the grace period elapses without a ping, the monitor turns **Unhealthy** and Pulse opens an incident named after the monitor. From there the incident follows its escalation policy like any other, so the right people get notified through the usual channels.

The next ping that arrives recovers the monitor and resolves the incident automatically.

Activity Feed

Pings are recorded in the monitor's activity feed, so you can answer "did the backup actually run every night this month?" without leaving Pulse. A run of consecutive pings is collapsed into a single entry -- click **Show all N** to expand it.

Pings that arrive less than 30 seconds apart are recorded once. A job that retries in quick succession therefore shows a single entry rather than one per attempt. This never affects the monitor's health: every ping resets the deadline, whether or not it adds a feed entry.

Because a heartbeat monitor can ping as often as once a minute, ping entries are kept for your organization's data retention window and then removed. Status changes -- the monitor going unhealthy and recovering -- are kept as the long-term record.

Notes

- Heartbeat monitors cannot be created through CSV import. Each one carries its own secret URL that has to be distributed to the job that pings it, so they are created one at a time.
- Heartbeat monitors can be included in scheduled reports by selecting **Heartbeat** in the report's monitor-type filter.
- See Monitors for filtering, muting, tagging, and everything else that applies to every monitor type.

4.6 Anomaly Detection

A threshold monitor needs you to know the number. Anomaly detection does not: Pulse measures what a data point normally does, then watches for it doing something else.

That is useful exactly where thresholds are not. Nobody knows the right high limit for four hundred tags, and the ones worth alarming on are often the ones nobody thought about. A data point that has quietly drifted, seized, or gone quiet is a fault whether or not anyone wrote a limit for it.

What you get is not a new kind of monitor. Each check Pulse creates is an ordinary condition monitor -- you can open it, read the logic, change it, or delete it like any other.

Setting It Up

1. Open the monitor for the data point you want watched
2. Open the **Actions** menu and choose **Detect anomalies**
3. Pick a sensitivity, choose which checks to create, and confirm

Pulse works out which checks are possible for that particular data point and offers those. Anything it cannot do, it says so and why -- see When a check is not offered.

Start with them switched off

The **Start these running** checkbox is off by default, and leaving it off is the cheapest way to find out whether the checks suit your plant.

With it off the checks run and record what they would have done, without notifying anybody. Come back in a week and look at what would have fired. If it looks right, switch them on from the monitor list; if it looks noisy, lower the sensitivity or delete them. Nobody gets paged while you decide.

The Four Checks

Check	Fires when	Needs a baseline
Stopped reporting	No fresh reading for 15 minutes	No

Check	Fires when	Needs a baseline
Outside its normal range	The reading leaves the range the data point normally sits in, and stays out	Yes
Stopped changing	Still reporting, but the value has not moved	Yes
Moving unusually	Swinging or ramping far more than the data point normally does	Yes

Stopped reporting is the one that always works. It needs nothing measured, so it is available on a data point you added this morning -- and it is what makes the others' silence meaningful. A data point whose band could not be built is still watched for going quiet, so "nothing is alerting" and "nothing is looking" stay different states.

Stopped changing is not the same as **Stopped reporting**, and the difference matters when you are woken up by one of them. A stuck value means the data point is still talking and the number behind it is frozen: a seized valve, a frozen scan cycle, a cached read. The device is fine and the measurement is not.

Moving unusually covers a fast ramp and a noisy oscillation with one check. Both are the same fact about the window -- it covered far more ground than usual -- and separating them would need a direction that the check deliberately ignores.

Sensitivity

Setting	Meaning
Low	Only the obvious. A wide band and a long hold before it fires
Medium	The recommended starting point
High	Catches things earlier, and fires more

Sensitivity changes two things together: how far outside normal counts as far, and how long it has to stay there. Both are deliberate -- an excursion that lasts ten seconds is usually the process, and one that lasts fifteen minutes usually is not.

High is not a fleet setting

High is a reasonable choice for one data point you are already worried about. Across a few hundred data points it produces incidents faster than anyone reads them, and a queue nobody reads is the same as no monitoring at all.

You can change the sensitivity of a check after it exists by opening its condition monitor and editing the numbers directly.

What Pulse Measures

Every night, Pulse measures each data point's recent history and stores what it found: the level the data point sits at, how far it normally strays from that, how much it moves within a few minutes, and how coarse its smallest real change is.

Three things follow from that, and they are worth knowing before you go looking for a check that is not there:

- **A new data point has no baseline.** The checks that need one become available after the next nightly measurement. **Stopped reporting** works immediately.
- **A measurement is kept until a better one replaces it.** Pausing a monitor for three weeks of maintenance does not destroy its baseline.
- **A measurement that goes stale stops the check.** If the nightly measurement has not run for over a week, the checks that depend on it report **Unknown** rather than judging against a number nobody stands behind.

When a Check Is Not Offered

The dialog lists the checks it cannot create for that data point, with the reason. These are answers rather than errors:

Reason	What it means
Pulse has not measured this data point yet	Normal for a data point added today. Available after the next nightly measurement
Not enough history yet to say what normal is	The data point has data, but too little of it
This data point is on/off	A normal range has no meaning for a two-valued signal. **Stopped reporting** is still offered

Reason	What it means
This data point does not report numbers	Nothing can be watched, including **Stopped reporting**
Does not vary in a way a normal range can be built around	A tag pinned at one value, or one that steps between a handful. A band around it would be either useless or constantly firing

The last one is the interesting refusal. Roughly half of all industrial tags sit in a control loop and barely move, and a band drawn around a signal that does not vary is a band that everything is outside of. Pulse would rather say a band will not work on that tag than create four thousand alerts.

Incidents

An anomaly incident says what was measured and what it was measured against, rather than "the condition was met":

> **Coolant Temp is outside its normal range** Measured 91.4, outside the learned normal range 76.8–82.4, continuously for 15 min.

> **Feed Pressure has stopped changing** Still reporting, but fixed at 4.02 for 30 min.

The duration is the hold the check was configured with, and it is a fact about the moment it fired rather than a running clock -- the incident list shows the ongoing age separately.

Incidents from these checks escalate through your escalation policy exactly like any other, and are acknowledged and resolved the same way.

An unevaluable check does not keep escalating

If the data point behind an open incident goes quiet, Pulse stops climbing the escalation policy rather than waking more people for something it can no longer confirm. The incident stays open, and escalation resumes by itself if the data point comes back.

The initial notification is never held this way -- you are always told the incident exists.

Limitations

Worth knowing before you rely on these checks:

- **No engineering units.** Pulse does not know that a tag is degrees or bar, so an incident quotes bare numbers.
- **No shift or time-of-day awareness.** A data point whose normal differs between a day shift and a night shift is measured as one population, which makes its normal range wider than either shift alone.
- **Cyclic signals are refused.** A tag that swings on a regular cycle is not something a static range describes, and Pulse declines rather than guessing.
- **Planned downtime is not recognised.** A plant shutdown looks like a fault to these checks. Pause the monitors, or expect them to fire.

4.7 Message Templates

A message template is the custom text Pulse puts in the incidents a monitor raises, instead of a generic one. It is written once on the monitor and used every time that monitor opens an incident.

Where to Set One

Monitor	Where the message is written
CVE monitor	The `incidentMessageTemplate` column when importing from CSV
Condition monitor	The **What happened** field on the condition's Alert node, in the condition editor

A heartbeat is a data source, and the condition it alerts through is a condition monitor like any other — its own edit page has no message-template field, but **Open in editor** on that page opens the condition editor, where the same Alert node **What happened** field applies.

A condition can carry several Alert nodes, each with its own incident name and description — so one condition can raise distinctly-worded incidents for different failure modes. See Conditions.

Writing the Message

The message is shown exactly as you write it, in the incident and in the notification that raises it. It supports Markdown for emphasis, lists, and links — see the Markdown Reference.

Writing effective messages

- Keep the message concise but informative.
- Name the equipment the incident is about, so a responder knows where to go without opening Pulse.
- Add context about what condition was violated and what to check first.
- Use clear, non-technical language for operators. :::

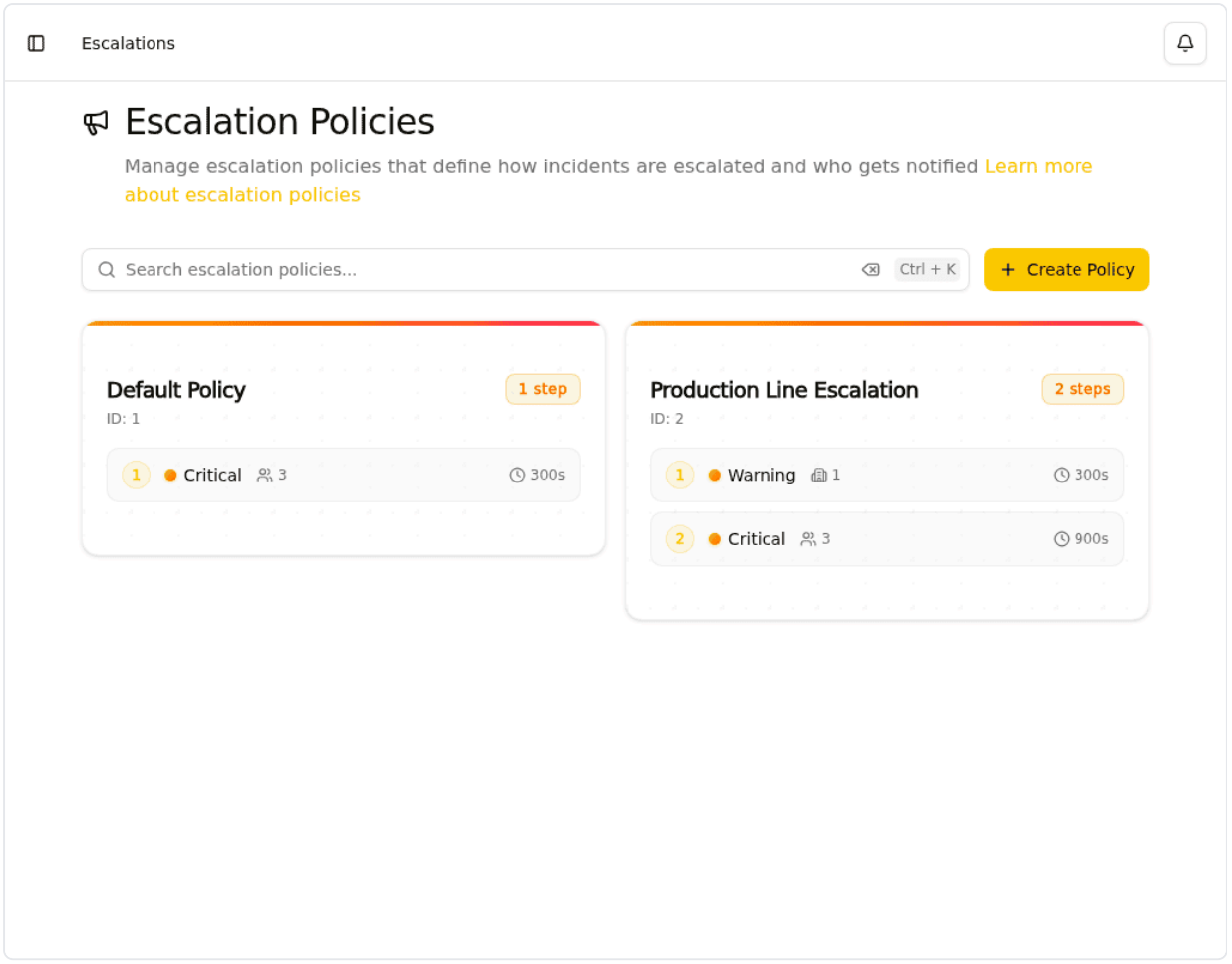
Related

- Markdown Reference
- Monitors
- OPC UA Configuration
- Siemens S7 Configuration

Part 5 · Alerting

5.1 Escalation Policies

Escalation policies define how incidents are communicated to your team. They specify who gets notified, through which channels, and in what order.



How Escalation Works

When an incident is created, Pulse follows the assigned escalation policy:

1. The first escalation step is triggered immediately (or after a configured delay)
2. Notifications are sent to the specified recipients via the severity's notification channels
3. If the incident is not acknowledged within the configured time, the next step is triggered
4. Steps can repeat a specified number of times before moving to the next level

When an **Operating Hours** step is reached outside its configured window, the escalation pauses and waits. The incident activity feed records an **Escalation paused — outside operating hours** event and shows when the escalation will resume.

Creating an Escalation Policy

1. Navigate to **Escalations** in the sidebar
2. Click **Create Escalation Policy**
3. Configure the policy name and steps

Configuring Steps

There are two types of steps:

Notify step — Sends notifications to the configured recipients.

Field	Description
Severity	The severity level that determines notification channels
Initial Delay	Seconds to wait before triggering this step
Repetitions	How many times to repeat this step before escalating
Recipients	Users, teams, and/or on-call schedules to notify

Operating Hours step — Pauses the escalation until the configured time window applies. When the escalation reaches this step outside the configured hours, it waits until the hours next begin before proceeding to the next step.

Field	Description
Mode	Weekdays, Everyday, or Custom (per-day schedule)
Hours	Start and end time of the window
Invert	When enabled, the gate holds while the hours apply and proceeds outside them

Operating hours are evaluated in the organization's timezone (configured in Settings).

Use an operating hours step to route escalation differently by time of day — for example, page the on-call team immediately but only escalate to the manager during business hours.

TIP

Start with a lower severity for the first step and escalate to higher severities. This prevents notification fatigue while ensuring critical issues reach the right people.

What Happens When Nobody Acknowledges

After all escalation steps have run, you can configure what Pulse does if nobody acknowledged the incident:

- **Don't repeat** -- Stop after the last step. No further notifications are sent.
- **Repeat N times** -- Restart the escalation sequence up to N times (maximum 20). Each full cycle runs all steps again.
- **Then escalate to** -- After the repetitions are exhausted, hand off to another escalation policy. This lets you chain policies — for example, repeating a team policy twice and then escalating to a management policy.

These settings are configured in the **What happens when nobody acknowledges** section of the escalation policy form.

The detail page for an existing policy shows a summary of this configuration.

When a Resolved Alert Comes Back

The **Reopen window** setting controls what happens when the same trigger monitor fires again shortly after an incident was resolved.

Setting	Behavior
0 (off)	Every re-fire starts a brand-new incident with its own responder and escalation from the first step
N seconds	If the same alert fires again within N seconds of being resolved, the original incident is reopened instead of a new one being raised

When an incident is reopened, it keeps its responder, comments, and escalation position — whoever already acknowledged it is not paged again. The resolved notification is also held back for the same duration, so a reopen within the window is never announced as fixed.

Set a reopen window if a flapping source keeps splitting one problem across several incidents. Set it to 0 to always start a fresh incident on each re-fire.

The policy detail page shows a summary of this setting under **When a resolved alert comes back**.

Example Configuration

A typical escalation policy might look like:

Step 1 -- Email notification

- Severity: Low
- Delay: 0 seconds
- Repetitions: 2
- Recipients: On-call team

Step 2 -- SMS notification

- Severity: Medium
- Delay: 300 seconds (5 minutes)
- Repetitions: 1
- Recipients: Team leads

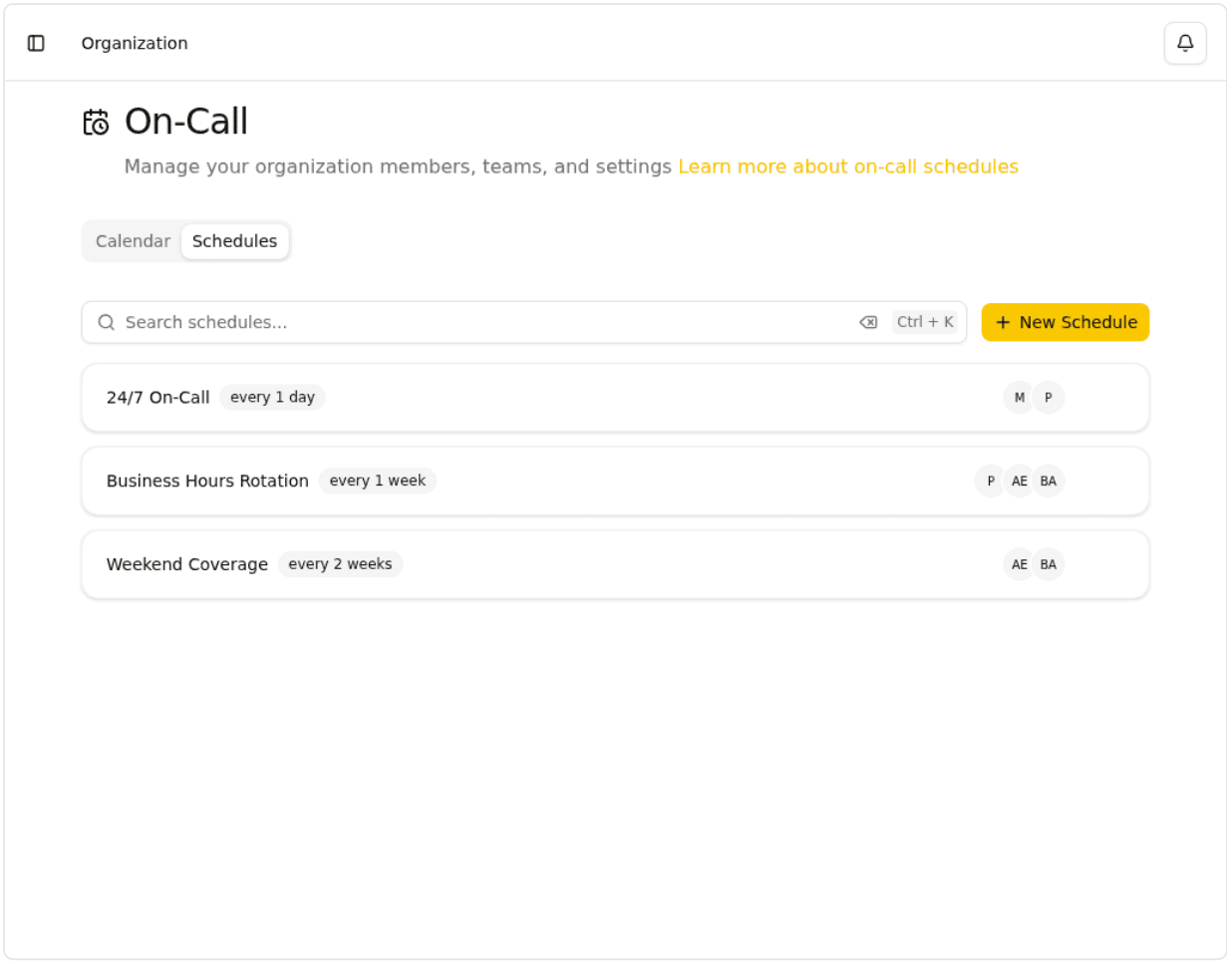
Step 3 -- Phone call

- Severity: High
- Delay: 600 seconds (10 minutes)
- Repetitions: 1
- Recipients: Plant manager

Part 5 · Alerting

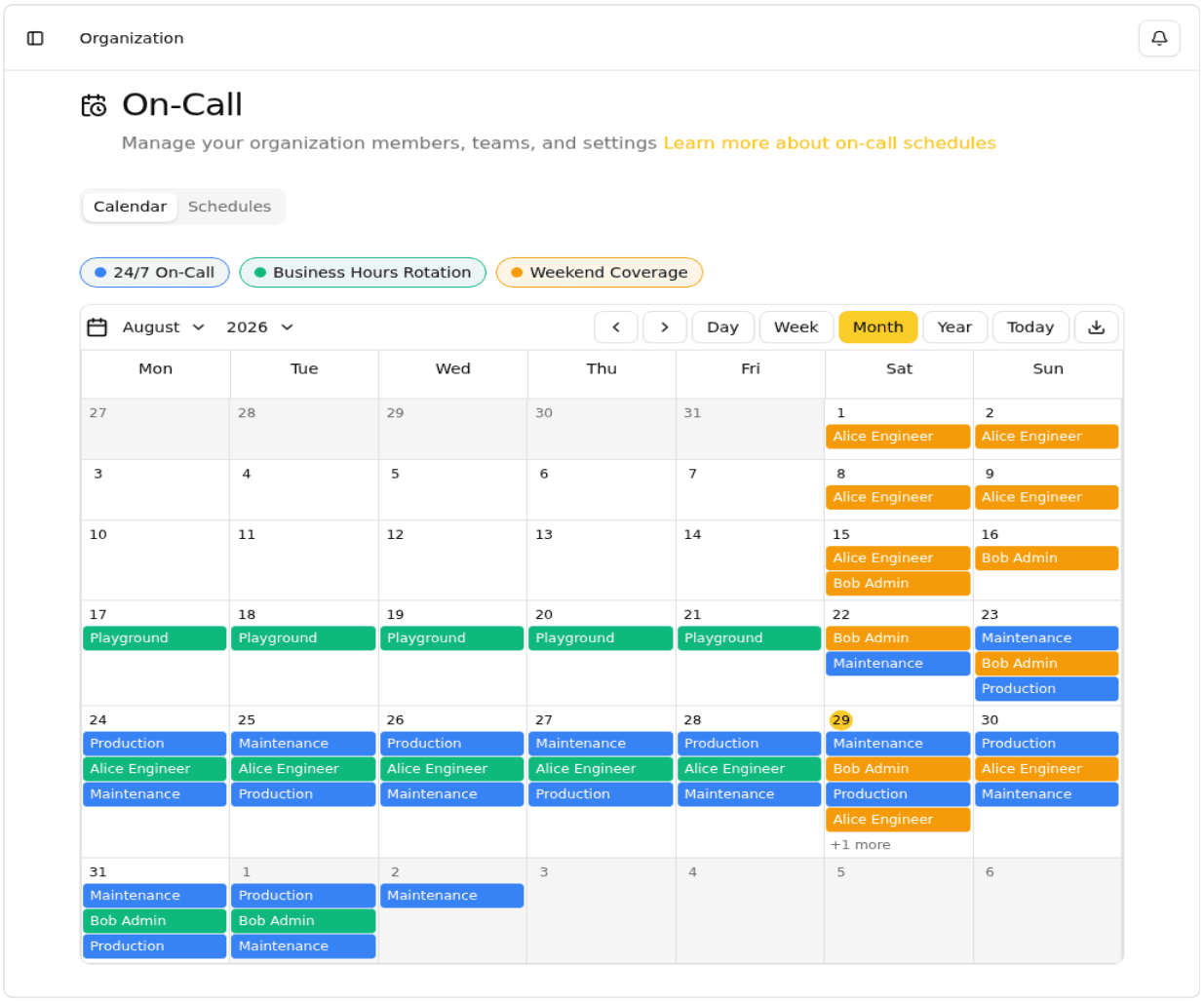
5.2 On-Call Schedules

On-call schedules define who is available to respond to incidents and when. Schedules are managed at the organization level under **Organization > On-Call** in the sidebar.



Calendar Tab

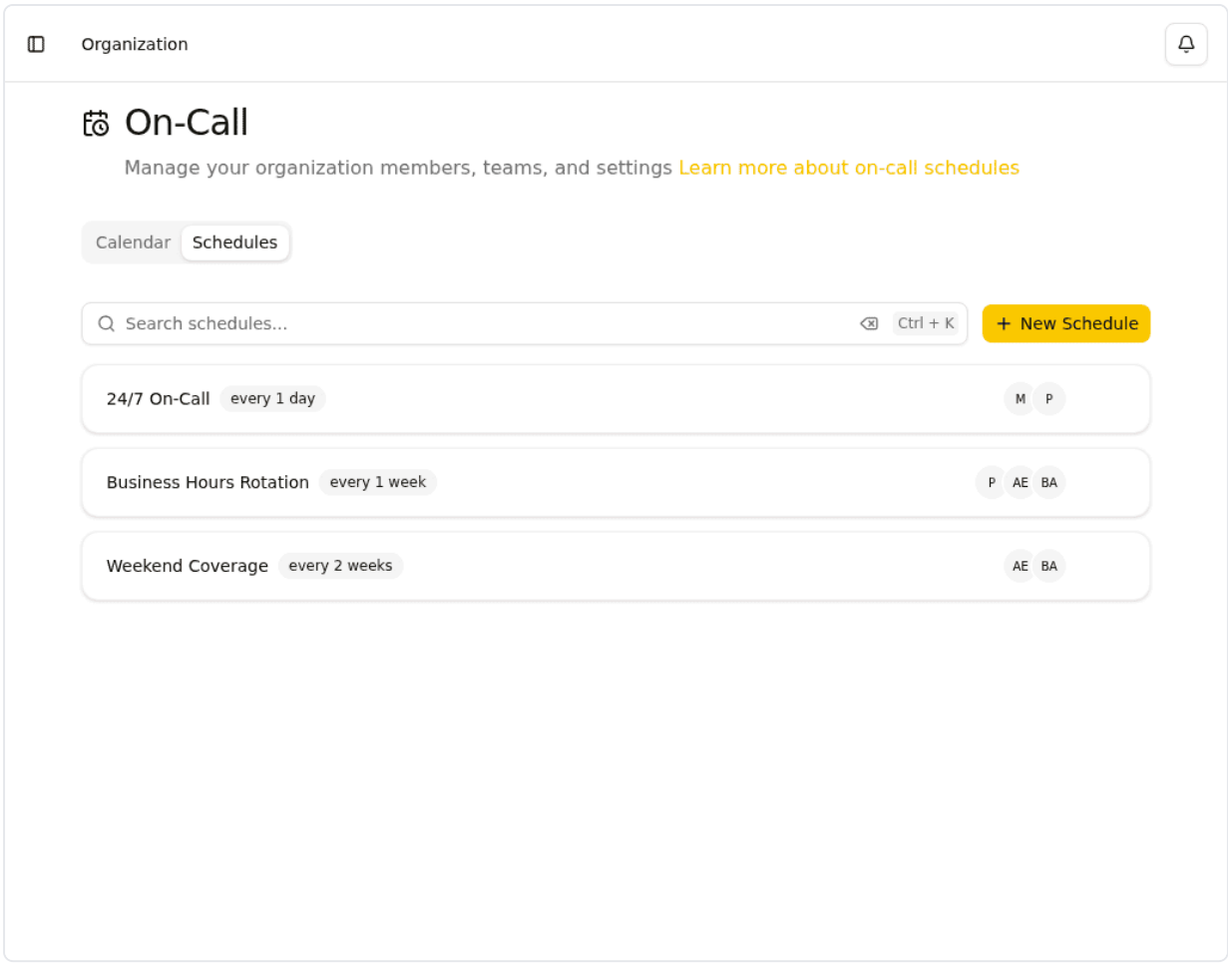
The **Calendar** tab provides a visual overview of all on-call schedules across your organization.



Use the view switcher to display the calendar in **day**, **week**, **month**, or **year** view. The selected view is reflected in the URL (e.g. `?view=month`), so you can share or bookmark a link that opens the calendar in a specific view. On mobile, a labeled **View** dropdown is shown above the calendar for easier switching; the mobile and desktop view preferences are stored independently, so switching to day view on your phone does not change your desktop preference.

Schedules Tab

The **Schedules** tab lists all on-call schedules. Each schedule shows its name, rotation interval, and assigned responders.



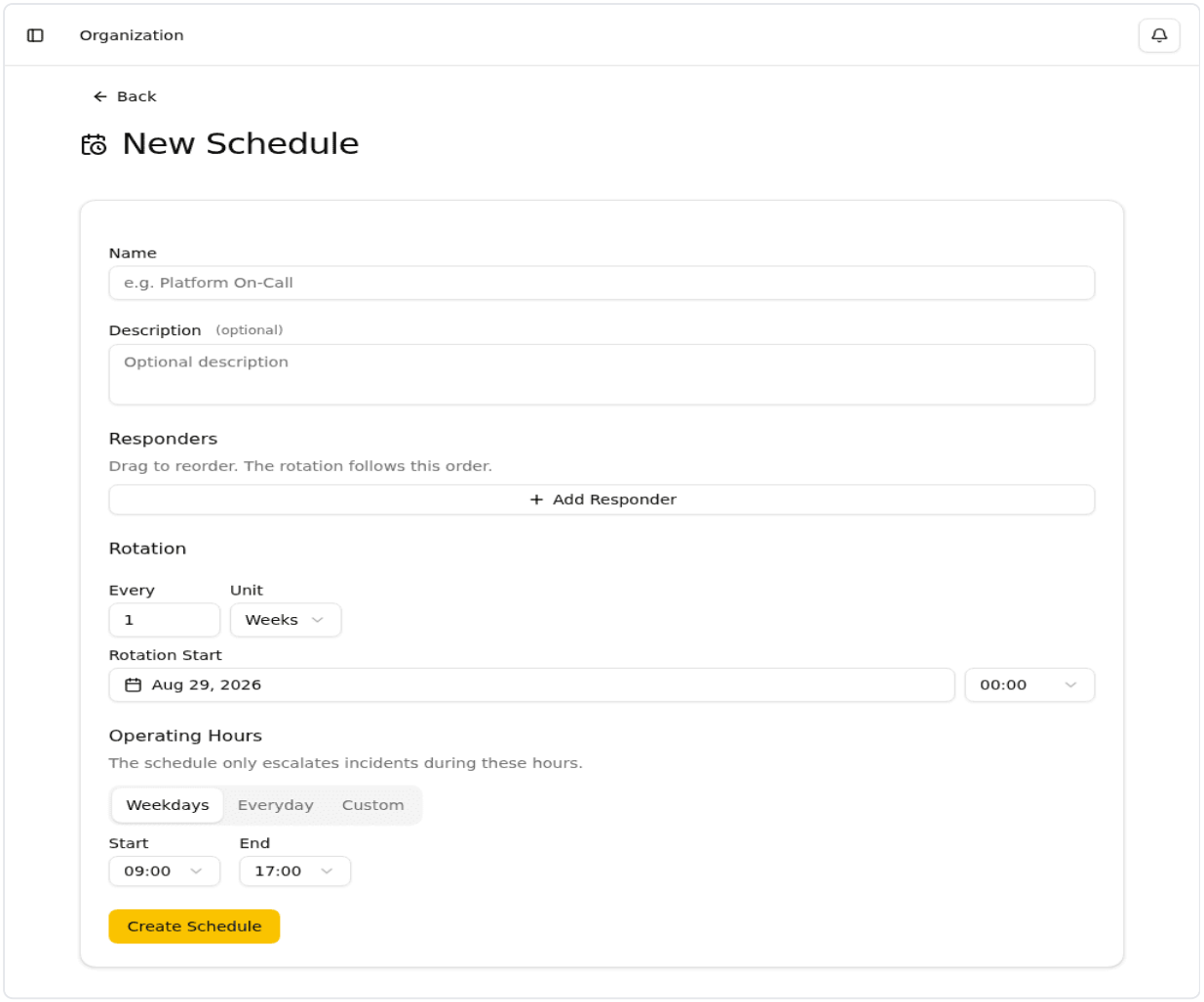
- **Search** -- Filter schedules by name
- **New Schedule** -- Create a new on-call schedule (Admin only)

Creating a Schedule

Click **New Schedule** to open the schedule form. Configure the following:

Field	Description
Name	A descriptive name for the schedule (e.g., "Platform On-Call")
Description	Optional description
Responders	Users or teams who participate in the rotation. Drag to reorder.
Rotation	How often the on-call responsibility rotates (minutes, hours, days, or weeks)
Rotation Start	When the first rotation begins

Field	Description
Operating Hours	Time window during which the schedule is active. Outside these hours, "nobody on call" alerts are suppressed.



TIP

The rotation follows the order of the responders list. Drag responders to change the rotation order.

TIP

Setting **Operating Hours** to `00:00–00:00` (start equals end) is treated as **all day** — the schedule is active for the full 24 hours.

INFO

When your on-call shift starts, Pulse notifies you by push notification and email: **"You are now on call for [schedule name]."** When your shift ends, you receive a second notification and email: **"Your on-call shift for [schedule name] has ended."** Notifications are sent in your preferred language. If you are on call for consecutive shifts, you will receive a start notification at the beginning of each shift. The email includes a direct link to the on-call calendar for that schedule.

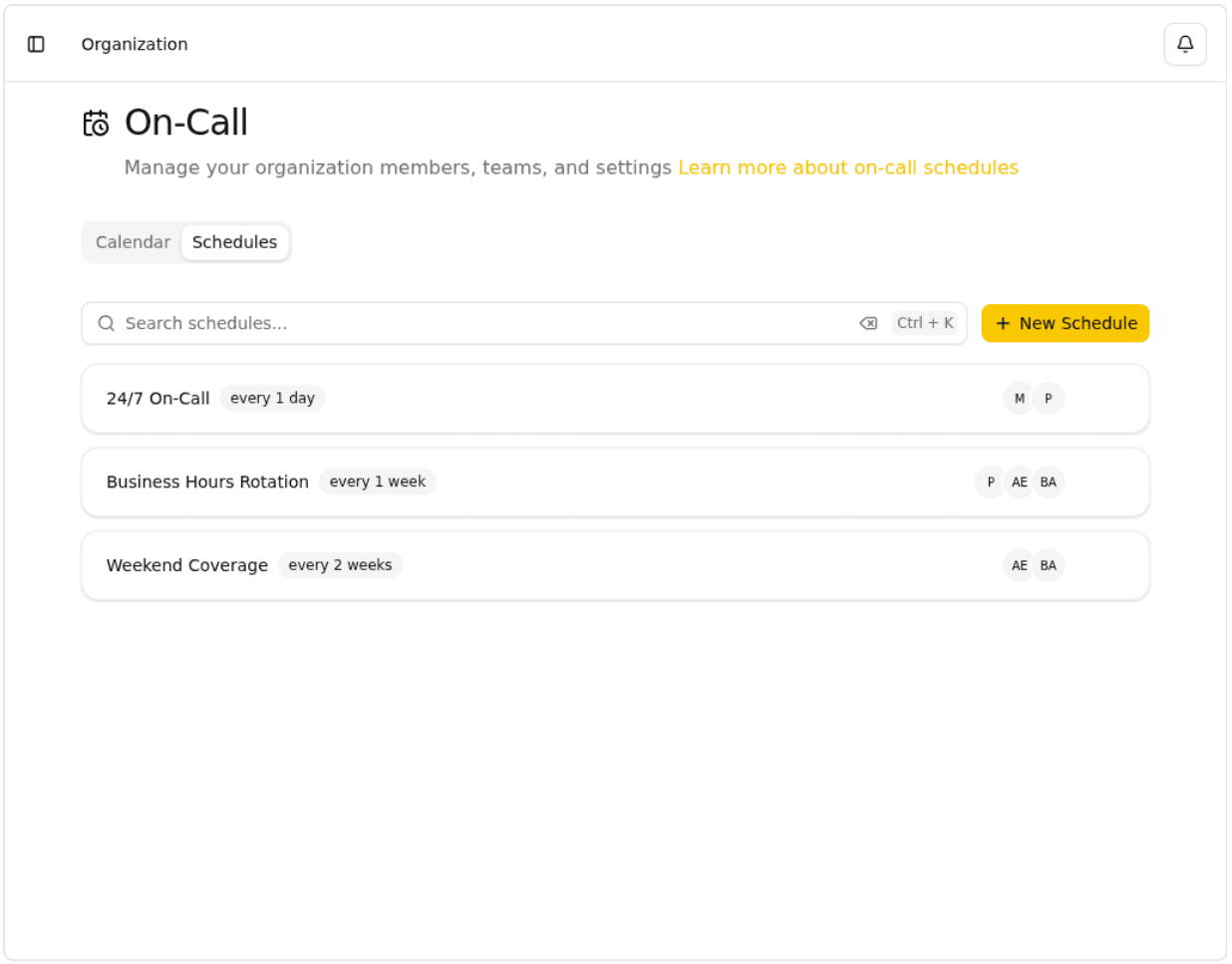
INFO

If no one is on call during configured operating hours, Pulse raises a system incident named **"No one on call for [schedule name]"** to alert your team. The schedule name is included so you can immediately identify which schedule has a coverage gap. Outside operating hours, this alert is suppressed — gaps in coverage are expected and not reported.

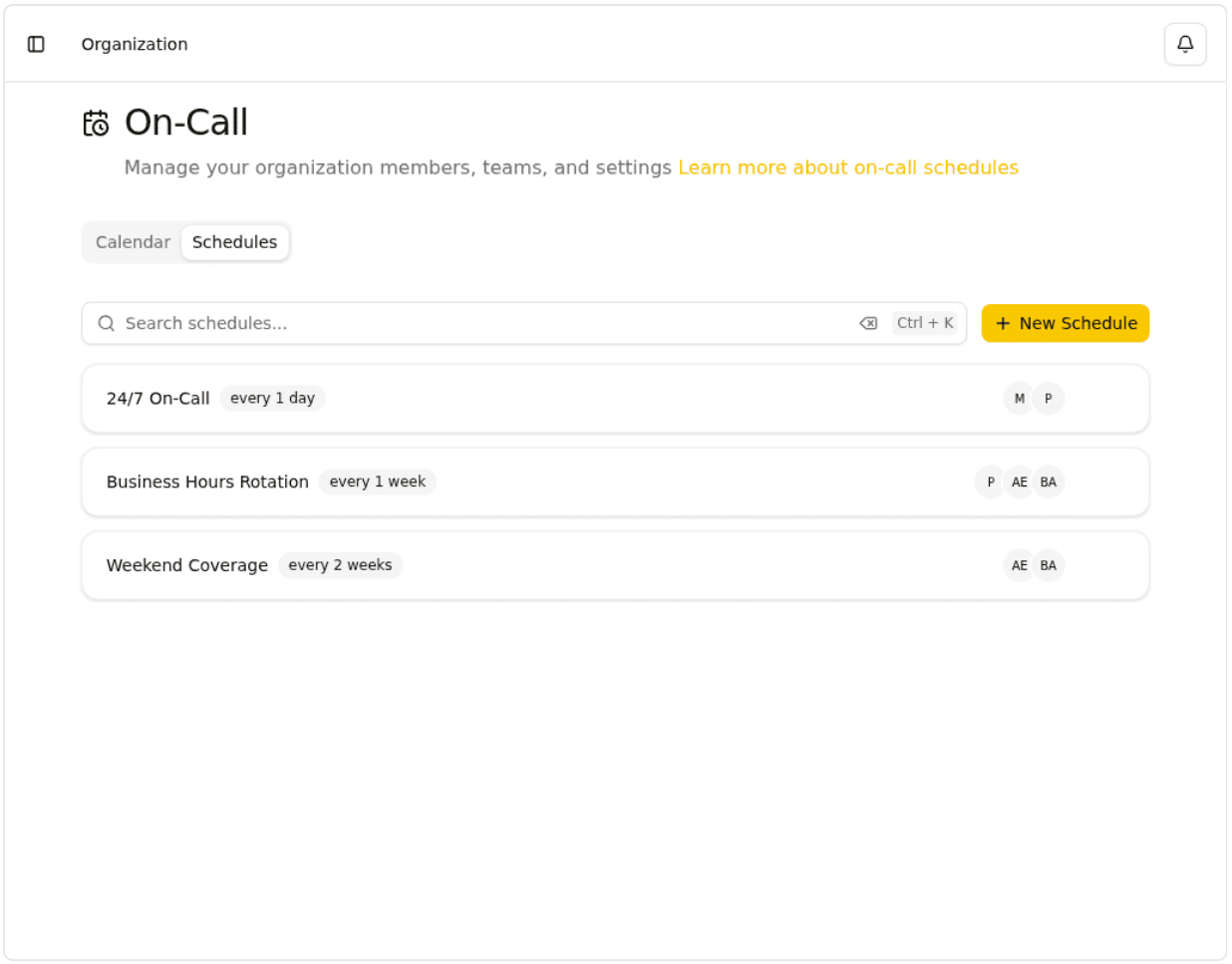
Managing Schedules

Click a schedule to view its detail page with two tabs:

- **Configuration** -- Edit the schedule name, responders, rotation, and operating hours



- **Overrides** -- Create temporary overrides to assign a different responder for a specific time period. Overrides are not restricted to the schedule's operating hours — they apply for the full time window you specify.



Override status badges:

- **Active** -- Currently in effect
- **Upcoming** -- Scheduled for the future
- **Past** -- Already expired

Override actions

The actions available for an override depend on where it sits in time:

Status	Available actions
Upcoming	Edit (Admin only), Delete
Active	Replace from now, End now
Past	None — past on-call can't be changed

Replace from now — Reassigns on-call responsibility from the current moment to the end of the active override. A dialog prompts you to select the new responder. The portion of the override that has already elapsed is preserved and unaffected.

End now — Ends the active override immediately. On-call returns to the normal rotation from this point. The elapsed portion is preserved.

> **INFO**
>
> Overrides that have already started or ended cannot be deleted. To stop an active override early, use **End now** or **Replace from now**.

> **INFO**
>
> The **Replace from now** and **End now** actions are also available from the calendar event popover. Click any override event on the calendar to access them.

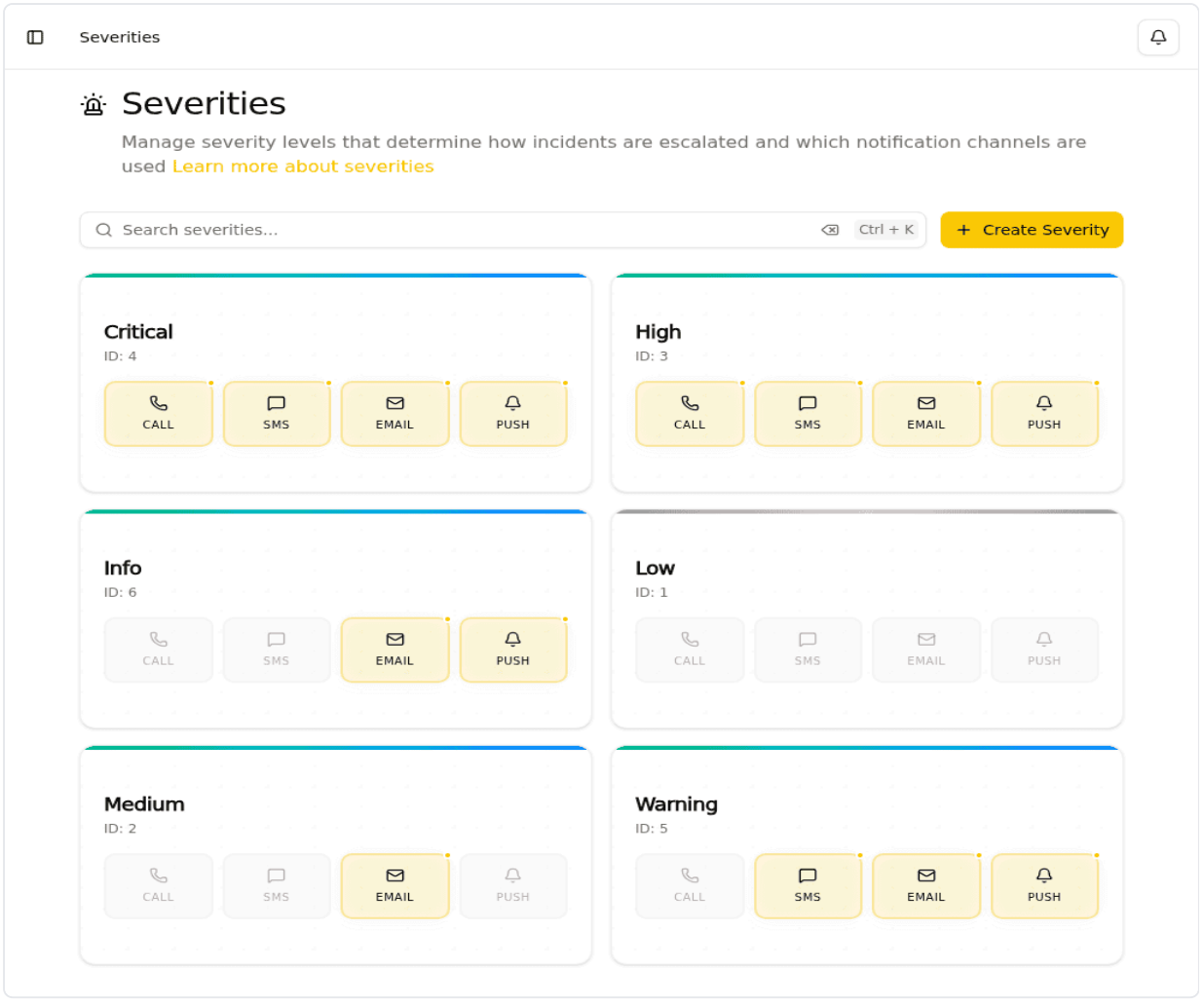
Related

- Escalation Policies -- Configure incident escalation
- Teams -- Manage team membership

Part 5 · Alerting

5.3 Severities

Severities define the importance level of incidents and determine which notification channels are used when escalating.



Notification Channels

Each severity level can be configured with one or more notification channels:

Channel	Description
Email	Send email notifications to recipients
SMS	Send text message notifications
Push	Send mobile push notifications

Channel	Description
Call	Initiate phone call notifications

Creating a Severity

1. Navigate to **Severities** in the sidebar
2. Click **Create Severity**
3. Enter a name and select the notification channels

> **TIP**
>
> Create severities that match your organization's incident response procedures. Common patterns include: Low (email only), Medium (email + SMS), High (email + SMS + call).

Using Severities in Escalation Policies

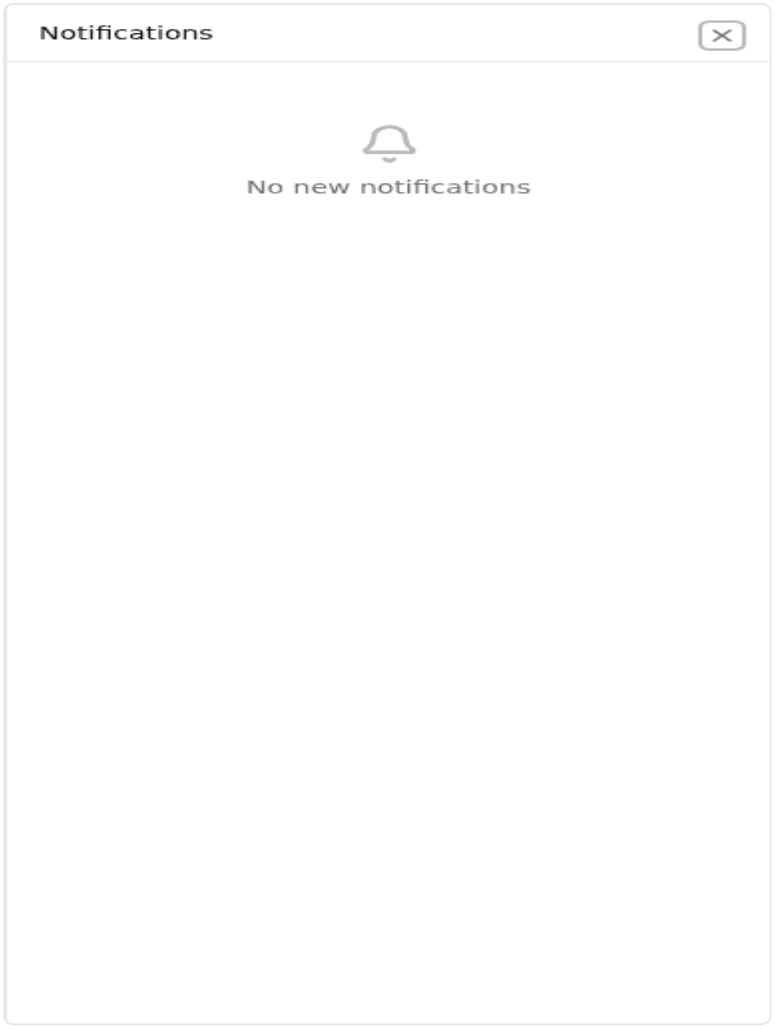
Severities are referenced in escalation policy steps. Each step specifies a severity level, which determines how recipients are notified at that escalation stage.

See Escalation Policies for details.

Part 5 · Alerting

5.4 Notifications

The notification center displays system alerts and important updates for your organization. It is accessible from the header bell icon on every page.



Opening the Notification Center

Click the **bell icon** in the top-right header to open the notification drawer. If there are unread notifications, a red badge with the count is shown on the icon.

Managing Notifications

Each notification shows:

- **Title** — a short description of the event

- **Details** — additional context (if available)
- **Time** — how long ago the notification was received

To dismiss a single notification, click the **×** button on the right side of the notification.

To dismiss all notifications at once, click **Dismiss all** at the bottom of the drawer.

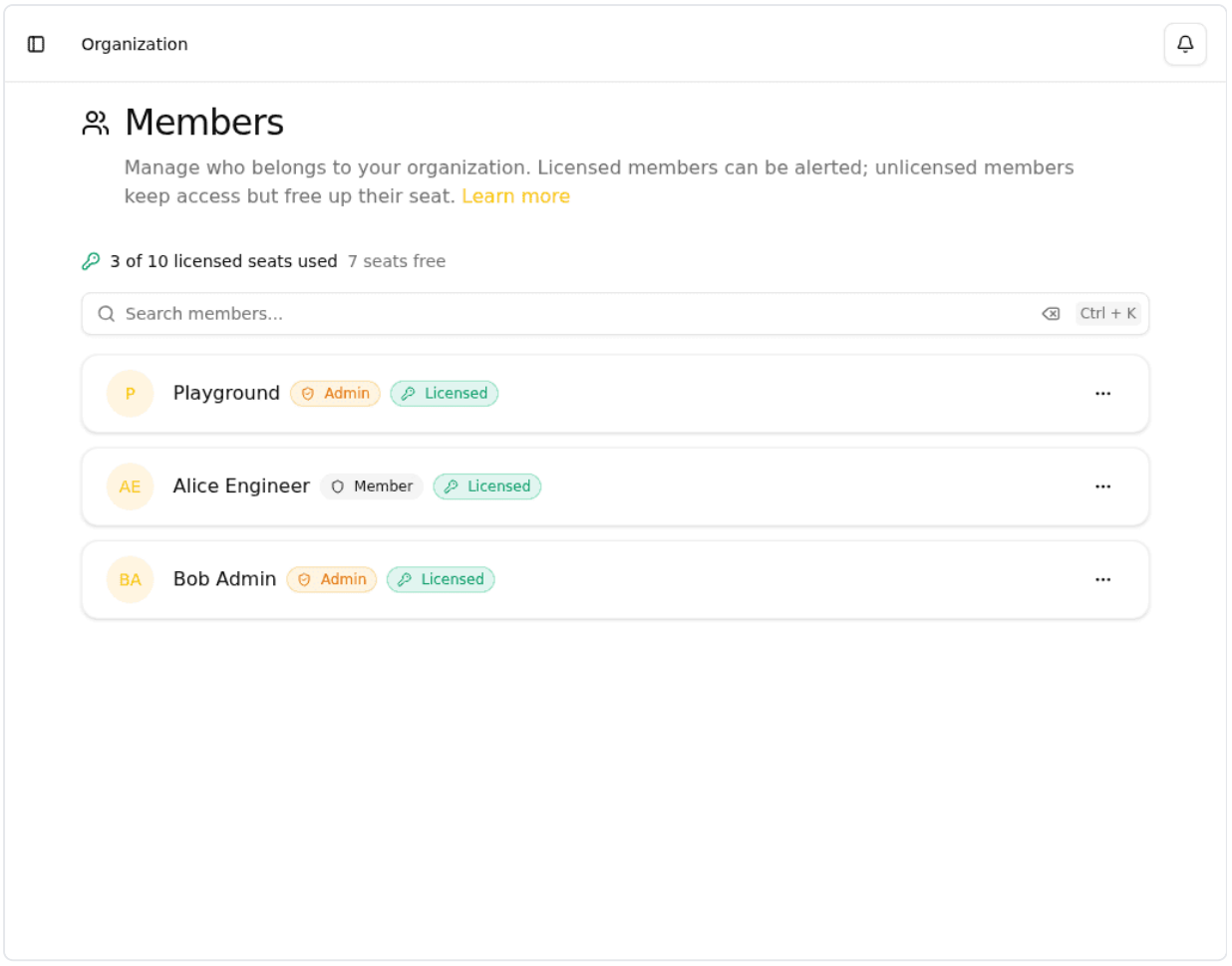
Related

- Incidents — View and manage active incidents
- On-Call Schedules — Configure who receives on-call notifications
- Mobile notifications — What the icons on push notifications mean

Part 6 · Administration

6.1 Members & Roles

Pulse uses a role-based access control system organized around organizations and teams. The **Members** page shows all users in your organization with their display names and roles.



- **Search** -- Filter the member list by name
- **Seat summary** -- Shows how many licensed seats are in use and how many remain
- **Role badges** -- Each member displays an **Admin** or **Member** badge
- **License badges** -- Each member displays a **Licensed** or **Unlicensed** badge

Organization Roles

Role	Capabilities
Admin	Create and manage monitors, tags, severities, escalation policies, and teams. Full configuration access.
Member	View monitors and incidents. Acknowledge and resolve incidents. View team information.

Managing Member Roles

As an admin, click the action menu next to a member to:

- **Promote to Admin** -- Grant full configuration access
- **Demote to Member** -- Restrict to view-only access
- **Remove Member** -- Remove the user from the organization

> **WARNING**
>
> Removing a member cannot be undone. The user will lose access to all organization resources immediately.

Licensed seats

Every member is either **licensed** or **unlicensed**. A licensed member holds one of your organization's licensed seats; the number of seats is set by your license.

State	What it means
Licensed	Holds a seat and can be alerted -- receives notifications and can be added as a recipient on escalation policies and on-call schedules.
Unlicensed	Keeps access to log in and view configuration, but frees up the seat and is never alerted. Cannot be added to escalation policies or schedules.

The members page shows how many seats are used and how many remain. When all seats are taken, assigning a license, adding a new member, or accepting an invite is blocked until you free a seat or raise the limit.

Changing a member's license state

As an admin, use the action menu next to a member to **Assign license** or **Remove license**. Removing a license is refused while the member is named directly as a recipient on an escalation policy or on-call schedule -- remove those references first.

Team Roles

Role	Capabilities
Team Admin	Manage team members and team settings
Team Member	Participate in the team and receive notifications

---

Inviting Users

Team admins can invite new users to their team:

1. Navigate to the team detail page
2. Copy the invite link
3. Share the link with the person you want to invite
4. The invited user can register a new account or log in with an existing one

> **TIP**
>
> Users who receive an invite link will be asked to either log in or create a new account before joining the team.

Creating an Organization

When you first register, you will be prompted to create an organization. You need to provide:

- **Organization Name** -- A descriptive name for your organization
- **Timezone** -- The default timezone used for date/time displays

Switching Organizations

If you belong to multiple organizations, you can switch between them using the user menu in the sidebar.

1. Click your name in the bottom-left corner of the sidebar
2. Select **Select Organization**
3. Choose the organization you want to work in

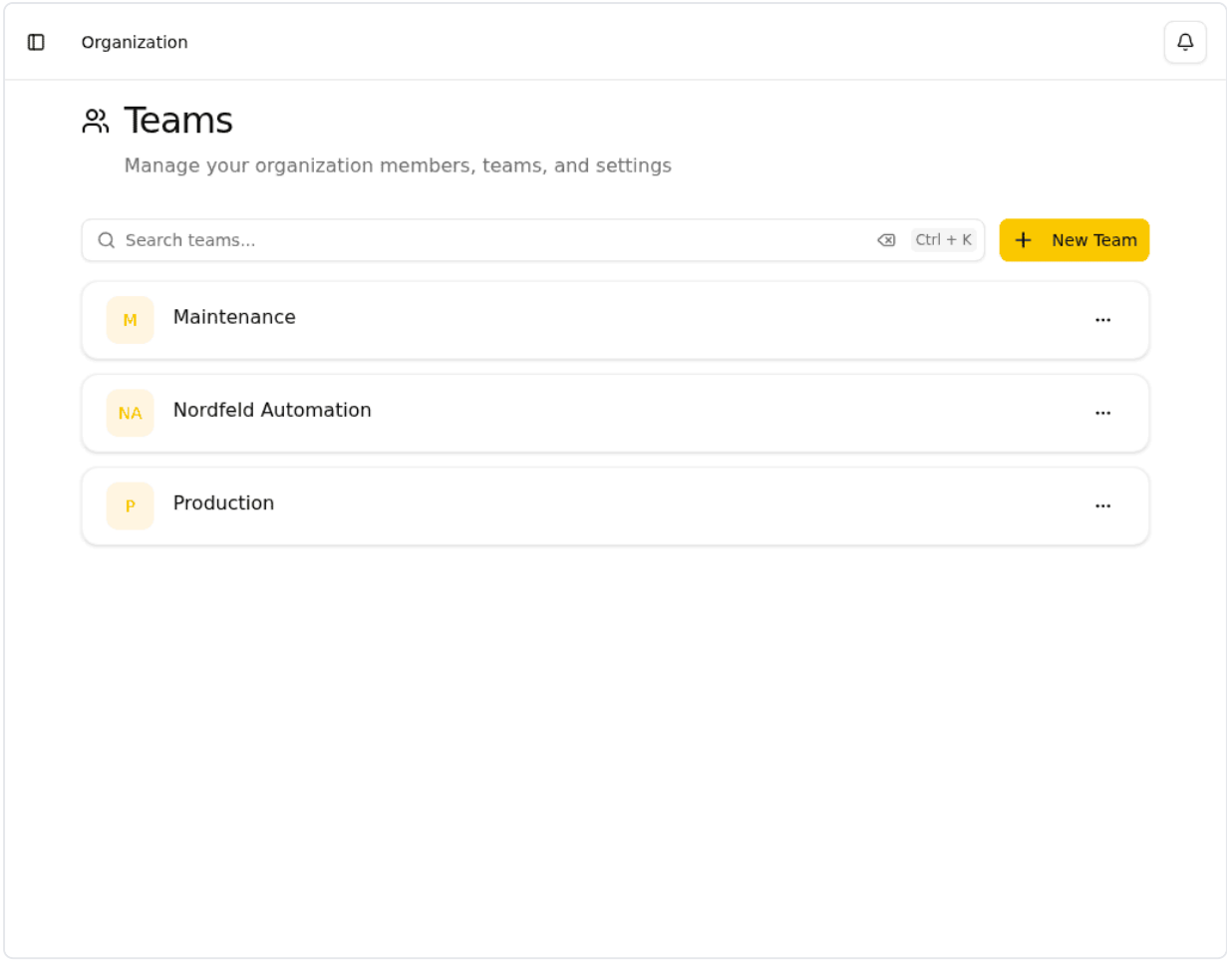
Related

- Teams -- Manage team membership
- Organization Settings -- SMTP servers and webhooks
- Licenses -- License keys, plan limits, and usage
- Preferences -- Personal notification settings

Part 6 · Administration

6.2 Teams

Teams group users together for notification routing and access control.



Creating a Team

1. Navigate to **Teams** in the sidebar
2. Click **New Team** (requires Admin role)
3. Enter a team name and description

Managing Team Members

On the team detail page, you can see all current members with their roles.

Inviting Members

1. Open the team detail page

2. Click **Copy Invite Link**
3. Share the link with the person you want to invite

The invited user will see an invitation page where they can accept the invite by logging in or creating a new account.

Teams in Escalation Policies

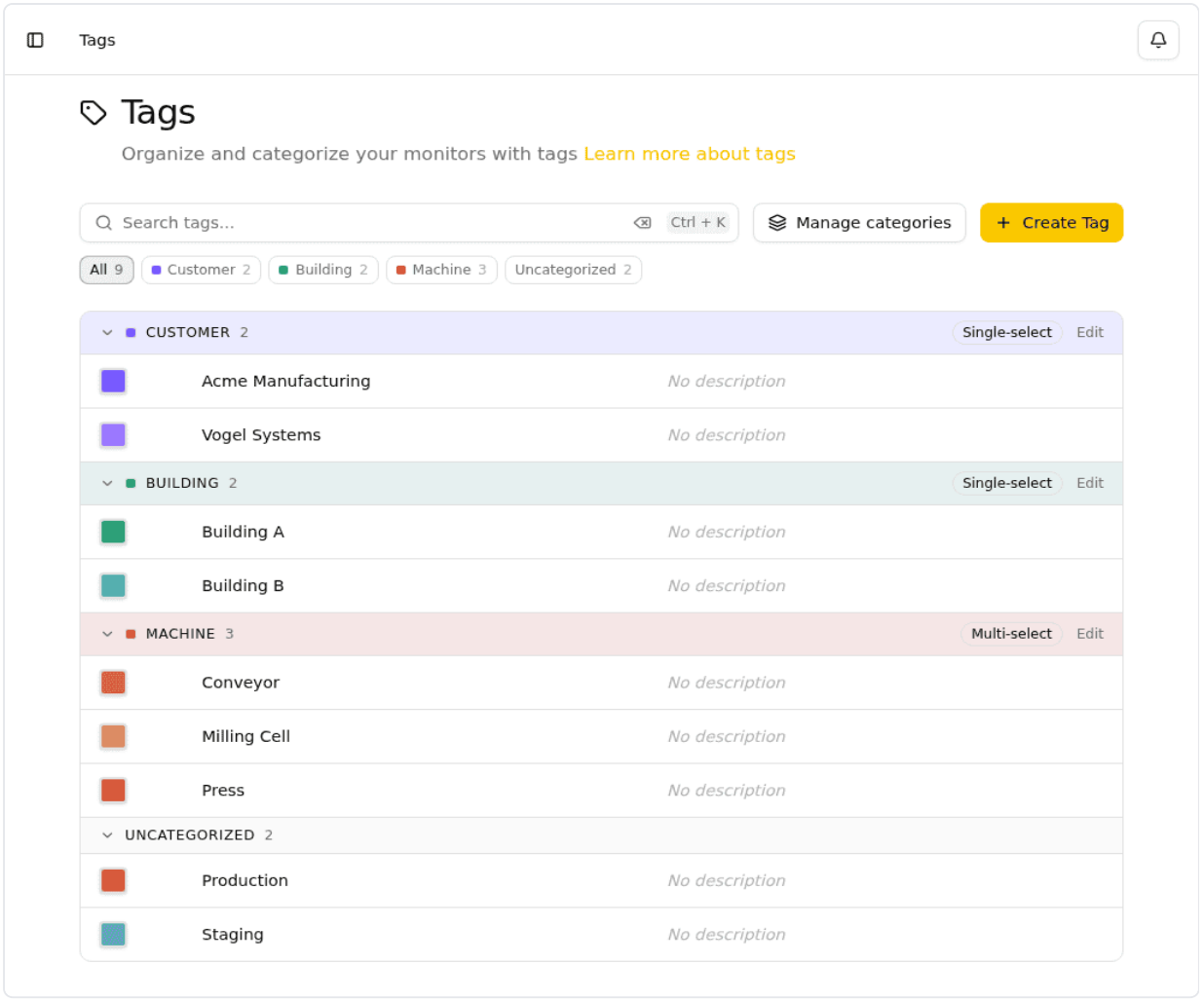
Teams can be added as recipients in escalation policy steps. When an escalation step targets a team, all members of that team who are within their operating hours will be notified.

See Escalation Policies for details on configuring team-based notifications.

Part 6 · Administration

6.3 Tags

Tags help you organize and categorize your monitors. You can assign tags to monitors and then filter the monitor list by tag.



Tag List

When tags belong to categories, the list is grouped by category. Each group shows the category name and tag count and can be collapsed or expanded by clicking the group header. A row of filter chips above the list lets you narrow the view to a single category. When you search by name, the list switches to a flat alphabetical view with a **Category** column so you can see at a glance which category each tag belongs to.

Creating a Tag

1. Navigate to **Tags** in the sidebar
2. Click **Create Tag**
3. Configure the tag:

Field	Description
Label	The display name for the tag
Description	Optional description of what this tag represents
Color	A color for visual identification
Category	Optional dimension this tag belongs to

Editing a Tag

Organization admins can edit an existing tag's label, description, and color. Click a tag in the list to open its detail page, make your changes, and click **Save Changes**.

Deleting a Tag

Organization admins can delete a tag from its detail page. Click the **Delete Tag** button, confirm the action in the dialog, and the tag is permanently removed.

> **WARNING**
>
> A tag that is still assigned to one or more monitors cannot be deleted. Remove the tag from all monitors first, then delete it.

Using Tags

After creating tags, you can assign them to monitors during creation or editing. On the monitor list page, use the tag filter to quickly find monitors with specific tags.

> **TIP**
>
> Use tags to group monitors by location (e.g., "Building A", "Line 3"), by criticality (e.g., "Safety", "Production"), or by equipment type.

Organizing Tags into Categories

Once you have more than a handful of tags, sort them into categories — Customer, Project, Site — so the tag picker groups them by dimension and filters can combine several dimensions at once.

See Tag Categories.

6.4 Tag Categories

A tag category is a dimension you sort your tags into — Customer, Project, Site, Line. Tags stay flat and independent; categories tell Pulse which tags belong to the same question, so you can filter your monitors along several dimensions at once.

Categories are optional. A tag without a category is listed as **Uncategorized** and keeps working exactly as before.

Why Use Categories

A plant with 400 monitors quickly ends up with 40 tags in one long, unsorted list — customer names next to line numbers next to criticality labels. Categories fix that in three ways:

- **Tags are grouped where you pick them.** The tag selector and every tag filter show one group per category, so "which customer is this?" and "which line is this?" are separate questions instead of one long list.
- **Filters combine across dimensions.** Selecting `Acme` from Customer and `Line 3` plus `Line 4` from Site answers "everything I run for Acme on lines 3 or 4" in one filter.
- **Some dimensions allow only one answer.** A monitor belongs to exactly one customer. Turn multi-select off for that category and picking a second customer replaces the first, so the mistake can't happen.

> **TIP**
>
> Good categories are the axes you would sort a spreadsheet by: Customer, Project, Site, Equipment Type. Things that describe a monitor's state or urgency — such as `Critical` — are usually better left uncategorized or handled with severities.

Creating a Category

Organization admins can create categories.

1. Navigate to **Tags** in the sidebar
2. Click **Manage categories**
3. Click **Create Category**
4. Configure the category:

Field	Description
Name	The dimension this category represents, such as Customer, Project, or Site
Description	Optional note on what belongs in this category
Color	A color for visual identification
Allow multiple tags per monitor	When on, a monitor can carry several tags from this category. When off, it can carry at most one

Category names must be unique within your organization.

Assigning Tags to a Category

A tag belongs to at most one category. Open a tag from the **Tags** list and choose a category in the **Category** field, or leave it on **Uncategorized**.

Tag labels only have to be unique within their category. You can have a `Line 3` under Site and a `Line 3` under Project without a conflict.

Single-Select vs. Multi-Select

The **Allow multiple tags per monitor** switch decides how the tag picker behaves for that category:

- **Multi-select** (the default) — a monitor can carry any number of tags from the category. Use it for dimensions that genuinely stack, such as Equipment Type.
- **Single-select** — picking a tag from the category replaces whichever tag from that category was selected before, the way a radio button works. Use it for dimensions where exactly one answer is correct, such as Customer.

Filtering Across Dimensions

Categories change how the tag filter on the monitor, incident, and vulnerability lists combines your selection:

- **Within one category, tags are combined with OR.** Selecting `Line 3` and `Line 4` from Site matches monitors on either line.
- **Across categories, they are combined with AND.** Adding `Acme` from Customer narrows that result to Acme's monitors on lines 3 or 4.

This is what makes several dimensions usable at once: each category you add narrows the result, while each extra tag inside a category widens it.

Guided Assignment

Both the tag filter and the tag picker on a monitor rank tags by what is already in use. As soon as you select a tag, every tag that appears on the same monitors moves to the top of its group with a count of matching monitors. Tags that never occur together drop below and dim — they stay clickable, so you can always broaden the selection.

In practice: pick `Acme` under Customer, and the Site group immediately shows the sites Acme actually has, ahead of the ones they don't.

> **TIP**
>
> The counts also work with nothing selected — then each tag shows how many monitors carry it, which is a quick way to spot tags nobody uses anymore.

Editing and Deleting a Category

Organization admins can edit a category's name, description, color, and multi-select setting from its detail page.

Deleting a category does not delete its tags. The tags survive and become **Uncategorized**; the monitors they are assigned to keep them. Re-assign them to another category at any time.

> **WARNING**
>
> Deleting a category cannot be undone. Its tags lose their grouping and have to be re-assigned one by one.

Related

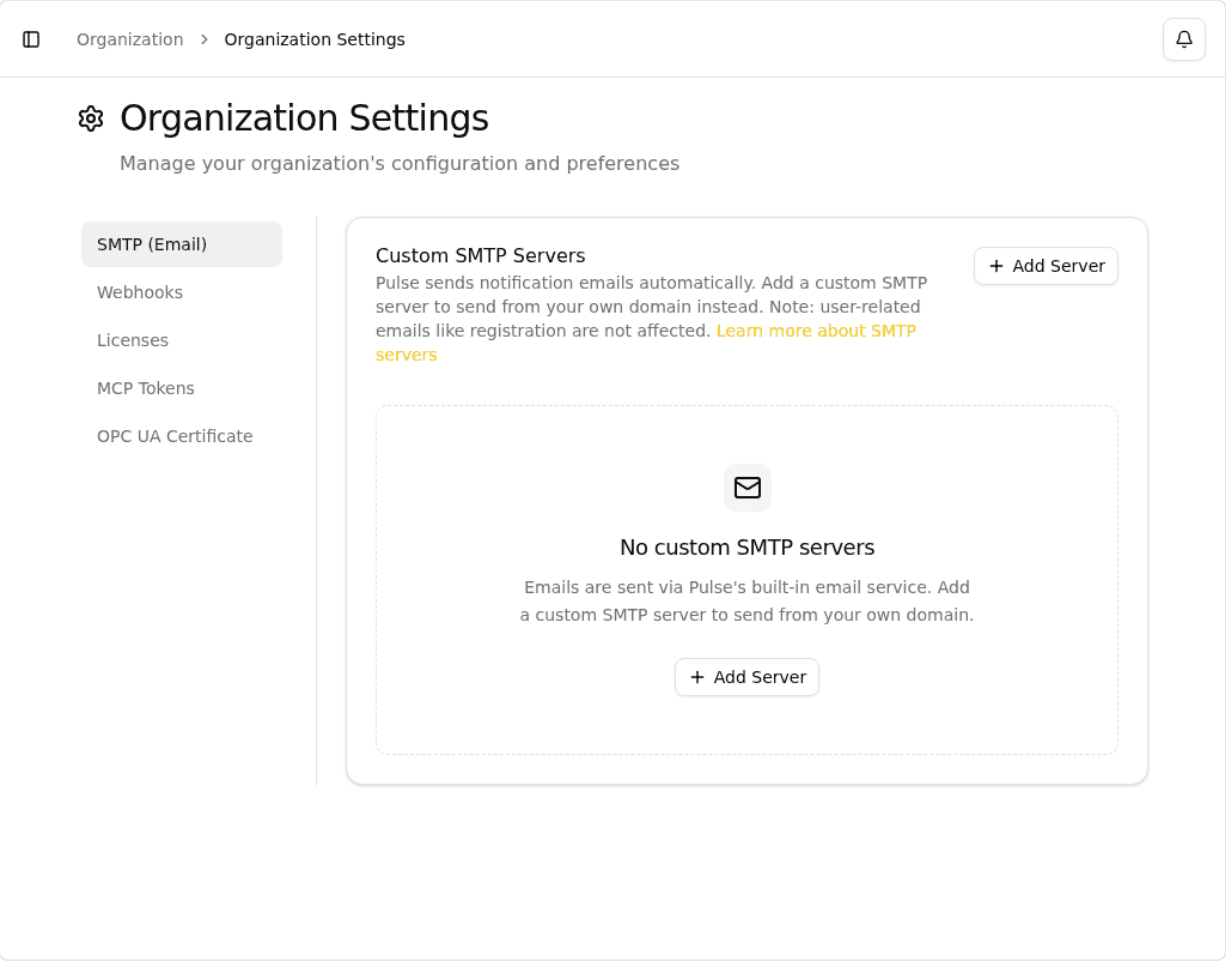
- Tags — creating, editing, and assigning tags
- Monitors — assigning tags while creating or editing a monitor

6.5 Organization Settings

Organization settings let admins configure infrastructure-level options. Navigate to **Organization > Settings** in the sidebar.

Custom SMTP (Email)

Configure custom SMTP servers for outgoing email notifications.



Pulse sends notification emails automatically using its built-in email service. Add a custom SMTP server to send emails from your own domain instead. When multiple custom servers are configured, Pulse tries them **in the order shown**, using the first one that succeeds.

Click **Add Server** to configure a new SMTP server:

Field	Description
Host	SMTP server hostname
Port	SMTP server port (e.g., 587 for TLS)
Username	Authentication username
Password	Authentication password
From Address	Email address used as the sender
From Name	Display name used as the sender
Use TLS	Enable TLS encryption for the connection

TIP

To validate SMTP settings you are able to send test e-mails

INFO

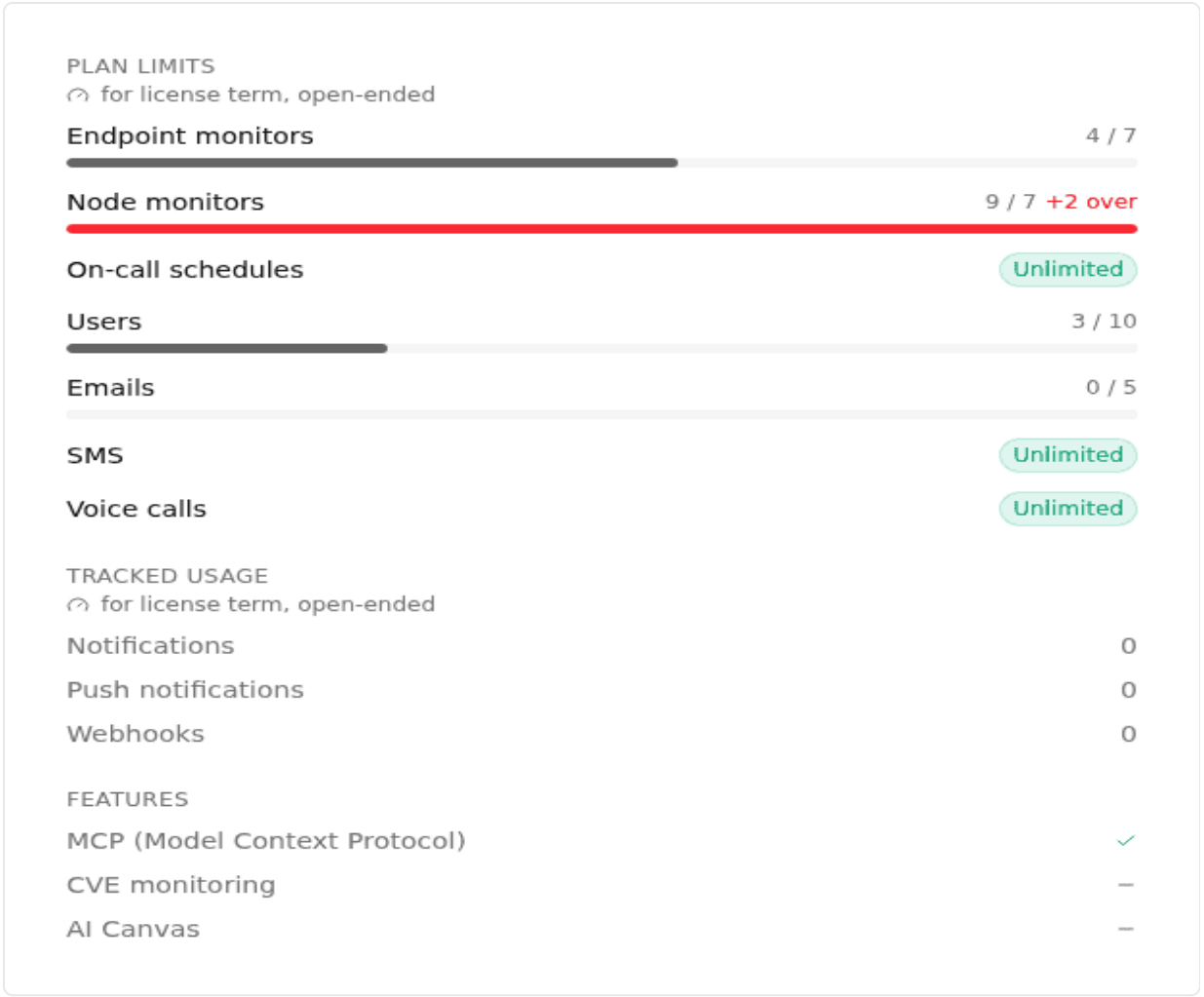
SMTP servers are used for notification emails only. User-related emails like registration are not affected.

Licenses

Licenses have their own page. See Licenses to add a license key, review your plan limits and usage, and learn about the license status banner.

License usage

The license usage panel shows how your organization's current consumption compares to the limits in your active license, measured over the license term.



The panel has two groups:

Plan limits — dimensions with a quota cap, shown as progress bars:

Dimension	Description
Endpoint monitors	Configured data sources against the plan limit
Node monitors	Kept data points against the plan limit
On-call schedules	Active on-call schedules against the plan limit
Emails	Emails sent during the license term
SMS	SMS messages sent during the license term
Voice calls	Voice calls made during the license term

Bar colors indicate capacity status:

- **Gray** — within limits (below 80 %)
- **Amber / near limit** — approaching the cap (80 – 99 %)

- **Amber / limit reached** — usage exactly at the cap (100 %)
- **Red / +N over** — quota exceeded

Dimensions not included in your plan show a **"not in your plan"** badge. If usage is recorded despite not being in the plan, the badge turns red.

Tracked usage — dimensions with no quota cap, shown as bare counts for the license term:

Dimension	Description
Notifications	In-app notifications sent
Push notifications	Mobile push notifications sent
Webhooks	Webhook deliveries made

Features — licensed capabilities shown as enabled (✓) or not included (—):

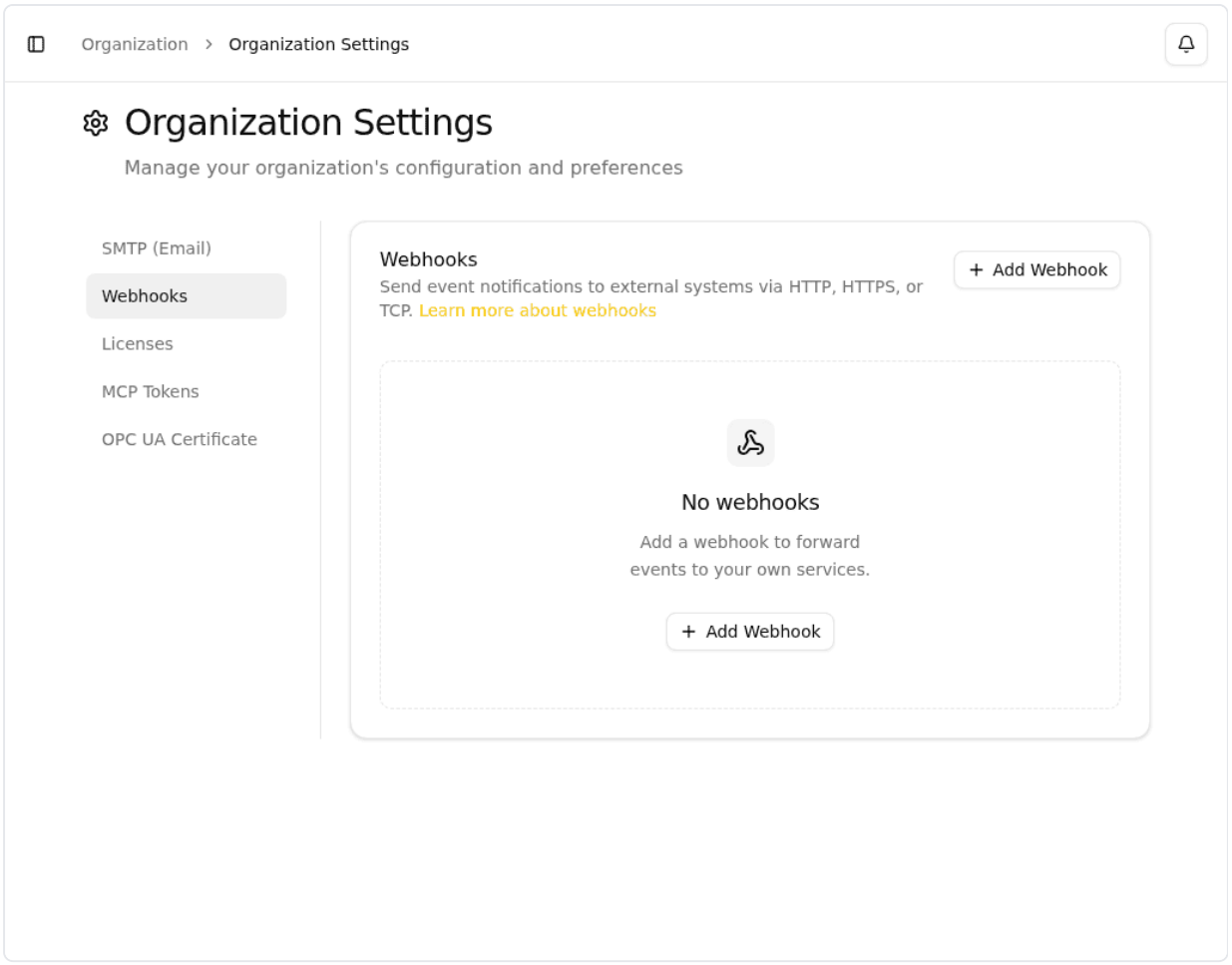
Feature	Description
MCP (Model Context Protocol)	Whether MCP clients can connect to this organization
CVE monitoring	Whether CVE monitors can be created and will scan
AI Canvas	Whether the AI Canvas investigation workspace is enabled

If no active license is attached to the organization, the panel shows an informational message instead. Contact your administrator to add a license key.

Webhooks

Webhooks let Pulse send real-time HTTP(S) or TCP notifications to external systems when events occur.

Navigate to **Organization > Settings > Webhooks** to manage webhooks.



Click **Add Webhook** to create a new webhook:

Field	Description
Display Name	A label to identify this webhook
Endpoint URL	Target URL — supports `https://`, `http://`, or `tcp://` schemes
Active	Toggle to enable or disable delivery
Event Types	Select which events trigger this webhook

WARNING

Pulse shows an "Unencrypted transport" warning when you enter a `tcp://` or `http://` URL. Plain TCP and plain HTTP do not encrypt payloads in transit — use them only for legacy systems that cannot use HTTPS.

INFO

TCP payloads are padded with spaces so the total message length (JSON + padding + newline) is always a multiple of **64 bytes**. This supports fixed-size stream buffers on SPS controllers that read in 64-byte blocks.

How delivery works

Each event triggers a delivery to every matching webhook. Deliveries to the same endpoint are processed **in order**, one at a time, so a slow endpoint never causes events to arrive out of sequence.

If a delivery fails — connection error, non-2xx HTTP response, or timeout — Pulse retries up to **3 attempts total** with exponential backoff between attempts. After the third failed attempt the event is dropped for that endpoint. For durability beyond this, point the webhook at a queue or reverse proxy you control.

HTTP(S) transport

HTTP(S) endpoints receive a `POST` with these headers:

Header	Value
`Content-Type`	`application/json`
`User-Agent`	`Pulse-Webhook/1.0`
`X-Pulse-Signature`	`sha256=<hex>` — only present if signing is on

The request body is the JSON event payload — verbatim, with no trailing newline.

TCP transport

TCP endpoints receive each frame in the order **JSON payload → ASCII-space padding (`0x20`) → a single newline (`\n`)** — that is, the spaces sit between the payload and the closing newline, which is always the final byte. The padding is sized so the **total frame length (payload + spaces + newline) is a multiple of 64 bytes**. Pulse opens a fresh TCP connection per delivery, sends the padded frame, then shuts down the write half of the socket and closes the connection.

NOTE

This is **raw TCP**, not WebSockets. There is no HTTP upgrade handshake, no opcode framing, and no persistent session — every event is its own short-lived connection. The 64-byte padding exists so PLCs and similar controllers can read in fixed-size blocks. The trailing newline and spaces are insignificant whitespace per the JSON specification (RFC 8259, where `JSON-text = ws value ws` and `ws` includes space `0x20` and newline `0x0A`), so any conformant JSON parser accepts the padded frame as-is — no trimming required. If your parser is stricter than the spec, read until end-of-stream and trim the trailing whitespace before parsing.

Verifying signatures

When you configure one or more **signing secrets** on a webhook, every HTTP(S) delivery carries an `X-Pulse-Signature` header containing an HMAC-SHA256 of the request body.

The header value is a comma-separated list of `sha256=<hex>` entries — one per signing secret, in the order the secrets are stored on the webhook. Multiple entries let you **rotate secrets without dropping deliveries**: add the new secret, deploy the new verifier (which accepts either secret), then remove the old secret.

To verify a delivery:

1. Read the **raw request body** as bytes — do not parse and re-serialize the JSON. Even insignificant whitespace differences will change the hash.
2. Compute `HMAC-SHA256(secret, body)` and format it as lowercase hexadecimal.
3. Compare your computed value `sha256=<hex>` to **any one** of the entries in `X-Pulse-Signature`. Use a constant-time comparison to avoid timing leaks.
4. If at least one entry matches, the delivery is authentic. If none match, reject the request.

Node.js example:

```
import crypto from "node:crypto";

// `body` must be the raw request body bytes, not a parsed object.
function verifyPulseWebhook(body, header, secrets) {
  const expected = secrets.map((secret) => {
    const hex = crypto.createHmac("sha256", secret).update(body).digest("hex");
    return `sha256=${hex}`;
  });

  const received = header.split(",").map((s) => s.trim());

  return received.some((r) =>
    expected.some(
      (e) =>
        e.length === r.length &&
        crypto.timingSafeEqual(Buffer.from(e), Buffer.from(r)),
    ),
  );
}
```

Python example:

```
import hmac
import hashlib

def verify_pulse_webhook(body: bytes, header: str, secrets: list[str]) -> bool:
    expected = [
        f"sha256={hmac.new(s.encode(), body, hashlib.sha256).hexdigest()}"
        for s in secrets
    ]
    received = [s.strip() for s in header.split(",")]
    return any(hmac.compare_digest(e, r) for e in expected for r in received)
```

WARNING

Always use a constant-time comparison (`crypto.timingSafeEqual` , `hmac.compare_digest` , or equivalent). A normal string `==` comparison can leak the expected signature one byte at a time via timing differences.

Event types

Each event type is identified by its `event.type` key in the webhook payload:

event.type	Description
incident.created	Incident created
incident.acknowledged	Incident acknowledged

event.type	Description
incident.resolved	Incident resolved
oncall.rotation	On-call responders or operating hours changed
notification.email_scheduled	Email notification scheduled
notification.sms_scheduled	SMS notification scheduled
notification.push_scheduled	Push notification scheduled
notification.call_scheduled	Voice call scheduled

Webhook payload envelope

Every webhook delivery is a JSON object. Pulse injects the `event_type` field at the top level alongside the event-specific fields:

```
{
  "event_type": "incident.created",
  ...event-specific fields
}
```

Incident event payloads

The `incident.created`, `incident.acknowledged`, and `incident.resolved` events share the same payload shape:

Field	Type	Description
event_type	string	One of `incident.created`, `incident.acknowledged`, `incident.resolved`
incident_id	integer	ID of the incident
title	string	Incident title
status	string	Incident status — `"created"`, `"acknowledged"`, or `"resolved"`

Notification event payloads

The `notification.email_scheduled`, `notification.sms_scheduled`, `notification.push_scheduled`, and `notification.call_scheduled` events share the same payload shape:

Field	Type	Description
`event_type`	string	One of `notification.email_scheduled`, `notification.sms_scheduled`, `notification.push_scheduled`, `notification.call_scheduled`
`incident_id`	integer	ID of the incident that triggered the notification
`channel`	string	Notification channel — `"email"`, `"sms"`, `"push"`, or `"call"`
`recipients`	array	List of notification recipients

Each entry in `recipients` has:

Field	Type	Description
`display_name`	string	Display name of the recipient
`email`	string|null	Email address, if available
`phone_number`	string|null	Phone number, if available

`oncall.rotation` payload

The `oncall.rotation` event fires whenever the set of active on-call responders changes or the operating-hours or override status of a schedule changes:

Field	Type	Description
`event_type`	string	Always `"oncall.rotation"`
`schedule_id`	integer	ID of the on-call schedule
`schedule_name`	string	Name of the on-call schedule
`previous_responders`	array	Responders active before this change
`current_responders`	array	Responders active after this change
`in_operating_hours`	boolean	Whether the schedule is currently within operating hours
`is_override`	boolean	Whether an override is currently active

Each entry in `previous_responders` and `current_responders` has:

Field	Type	Description
`id`	integer	Login ID
`name`	string	Display name or email
`email`	string	Email address
`phone`	string|null	Mobile number, if set

Connected MCP clients

AI assistants and automation tools connect to Pulse via the Model Context Protocol using OAuth: each user authorizes a client in the browser, per organization. There are no tokens to create or copy — the client discovers Pulse's authorization server, registers itself, and sends you to a consent screen.

> **INFO**
>
> MCP access requires an active license that includes MCP. The consent screen only offers organizations whose license includes MCP; if none of your organizations qualify, contact your administrator to upgrade the license.

Connecting a client

The MCP endpoint is available at `/mcp` on your Pulse instance. The **Connected MCP clients** settings tab shows the exact URL with a copy button.

> **Prerequisite**
>
> `CS_BASEURL` must be set to the externally reachable base URL of your Pulse instance (e.g. `https://pulse.example.com`). The OAuth discovery metadata and the authorization endpoint URL are derived from it — with the default value, clients on other machines cannot complete the flow.

Claude Code

```
claude mcp add --transport http pulse https://<your domain>/mcp
```

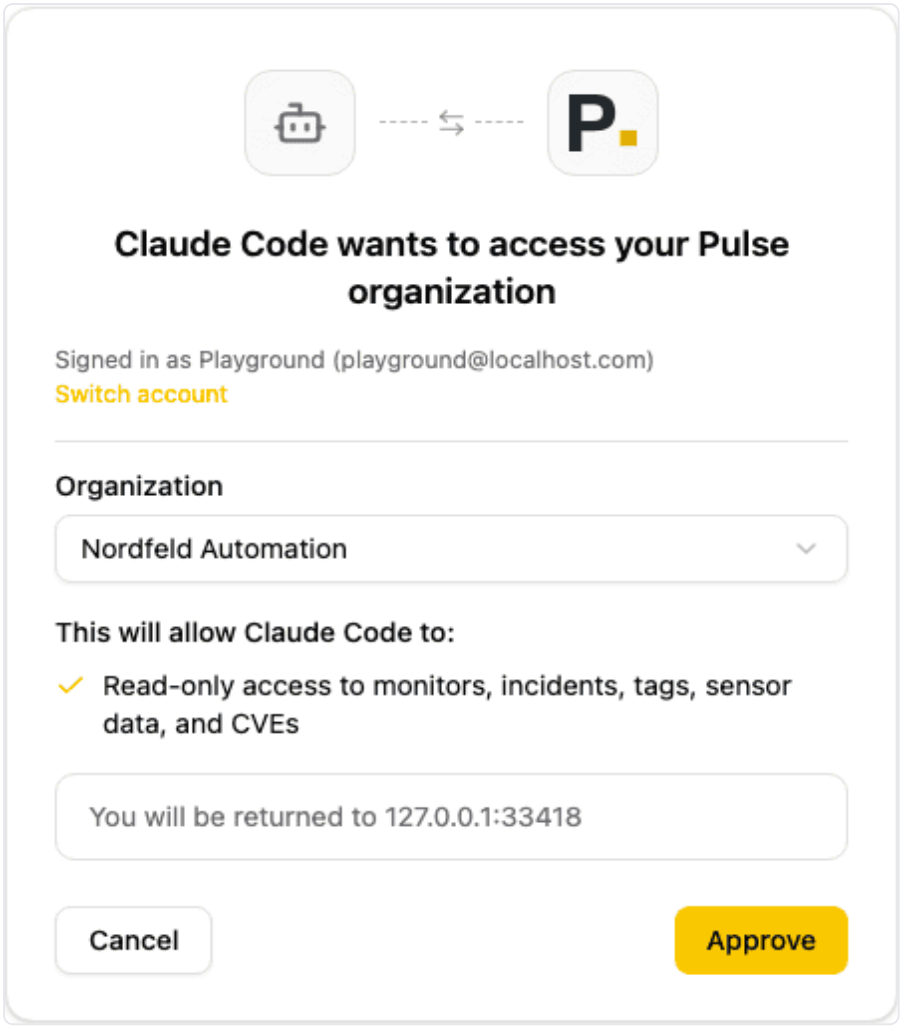
Replace `<your domain>` with the host your Pulse instance is reachable at — the same value as `CS_BASEURL`. Add `--scope user` to make the server available in every project on your machine instead of just the current one.

The first tool use (or `/mcp` in Claude Code) opens your browser: sign in to Pulse, pick the organization to connect, and approve. Tokens are handled automatically from then on.

claude.ai, Cursor, and other MCP clients

Add a remote (HTTP) MCP server with the URL `https://<your domain>/mcp`. The client discovers the OAuth configuration, registers itself automatically, and opens the Pulse consent screen in your browser.

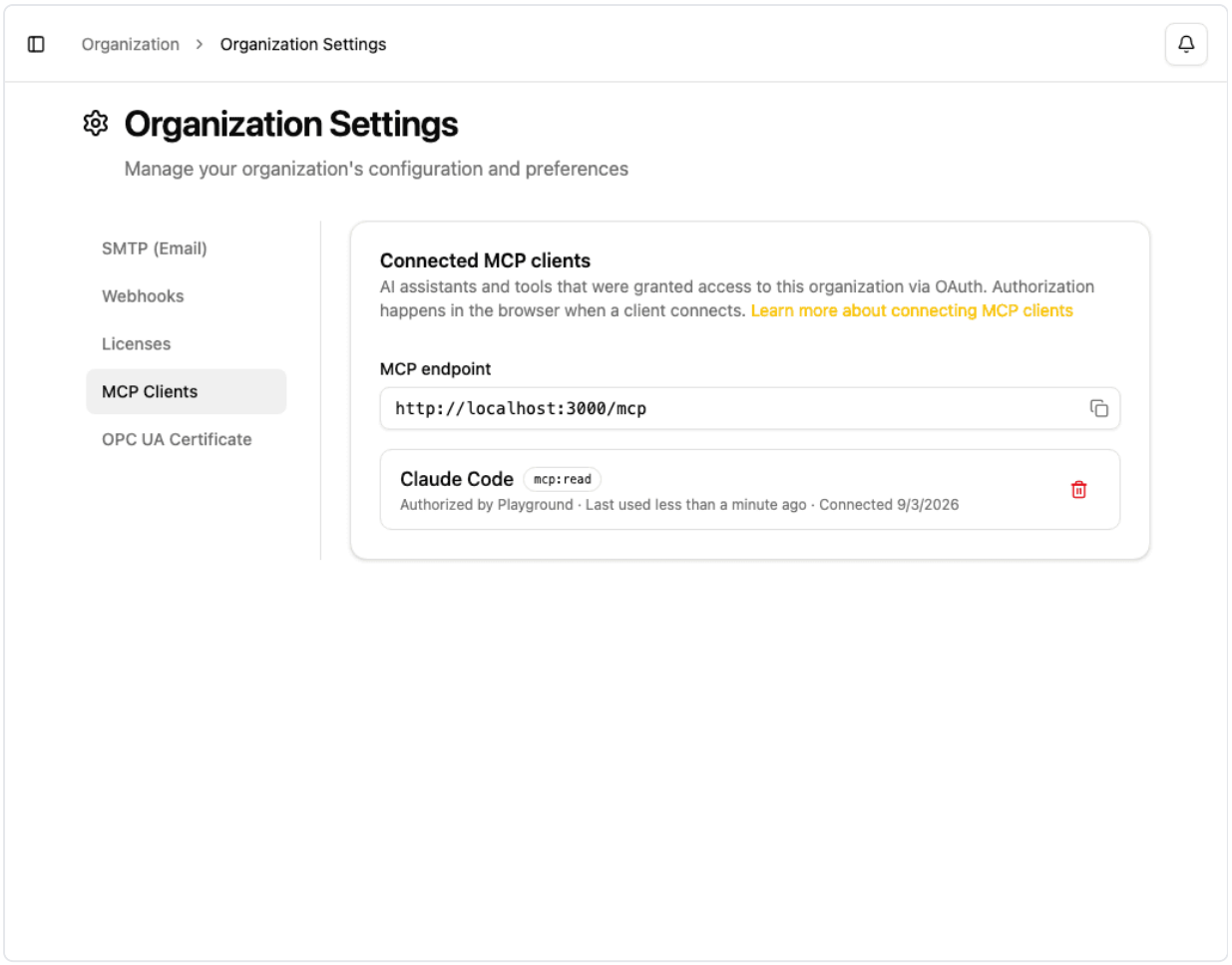
The consent screen



The consent screen names the client, lets you choose which organization to connect, and lists what access it gets (read-only access to monitors, incidents, tags, sensor data, and CVEs). It also shows where you will be returned after approving — verify that host matches the tool you started the connection from before clicking **Approve**.

Managing connected clients

Navigate to **Organization > Settings > MCP Clients** to see every client that was granted access to the organization.



Each row shows the client, who authorized it, when it was last used, and when it was connected. To revoke access, click the delete icon and confirm.

> **WARNING**
>
> Revocation is immediate — the client's next request is rejected, even if its current token has not expired yet. Revoke clients you no longer use.

Available MCP tools

The Pulse MCP server exposes these tools to connected clients. Parameters are self-describing — your AI assistant will discover them automatically via the MCP schema.

- `list_monitors` — List and filter monitors in your organization.
- `list_incidents` — List and filter incidents.
- `query_data` — Query time-series data with aggregations, displayed as an ASCII graph or table. Results can be grouped by monitor, by data point, or by data source.
- `query_cves` — Query CVE findings across your monitored devices.
- `list_tags` — List the organization's tags.

- `list_data_sources` — List data sources and, optionally, their individual data points: addresses, last readings, and which monitors read them. This is how an assistant finds the source names and point addresses that `query_data` filters on, and the only tool that can show a data point no monitor alerts on.
- `get_condition` — Read the condition graph a condition monitor alerts on, together with what every node in it currently evaluates to and when that value began. A condition monitor has no single threshold, so this is what answers which input tripped it.
- `list_incident_events` — Read one incident's timeline: when the monitor failed and recovered, every notification delivered and to whom, acknowledgements, resolutions, and escalation steps.

OPC UA Certificate

Navigate to **Organization > Settings > OPC UA Certificate** to manage the **application instance certificate** — the shared X.509 identity Pulse presents when opening secure OPC UA channels (Sign or Sign & Encrypt mode). See OPC UA Security — Application instance certificate for a conceptual overview and the full rotation workflow.

The tab shows the active certificate's subject, validity dates, and SHA-1 thumbprint. When a rotation is in progress, both the active and the pending certificate are shown. Each thumbprint is labelled — **Current** for the active certificate and **New — trust this before completing rotation** for the pending one — so the two are never confused.

Actions

Action	Available when	Description
Download PEM	Active certificate exists	Download the certificate file to install on your OPC UA servers
Copy thumbprint	Active certificate exists	Copy the SHA-1 thumbprint to cross-check in your server's certificate manager
Generate certificate	No active certificate	Provision the certificate ahead of time (Pulse also provisions it automatically on the first secure connection)
Rotate certificate	Active certificate exists, no rotation in progress	Mint a replacement to begin the overlap-window rotation

Action	Available when	Description
Complete rotation	Pending certificate exists	Promote the pending certificate to active — do this only after every server trusts the new thumbprint
Discard replacement	Pending certificate exists	Cancel the in-progress rotation and remove the pending certificate — a confirmation dialog appears
Revoke certificate	Active certificate exists	Emergency only: immediately replace the active certificate with a new identity

WARNING

Revoke certificate causes an immediate outage for all secure OPC UA monitoring — every server will reject Pulse until you trust the new certificate. Use rotation for a planned, outage-free replacement.

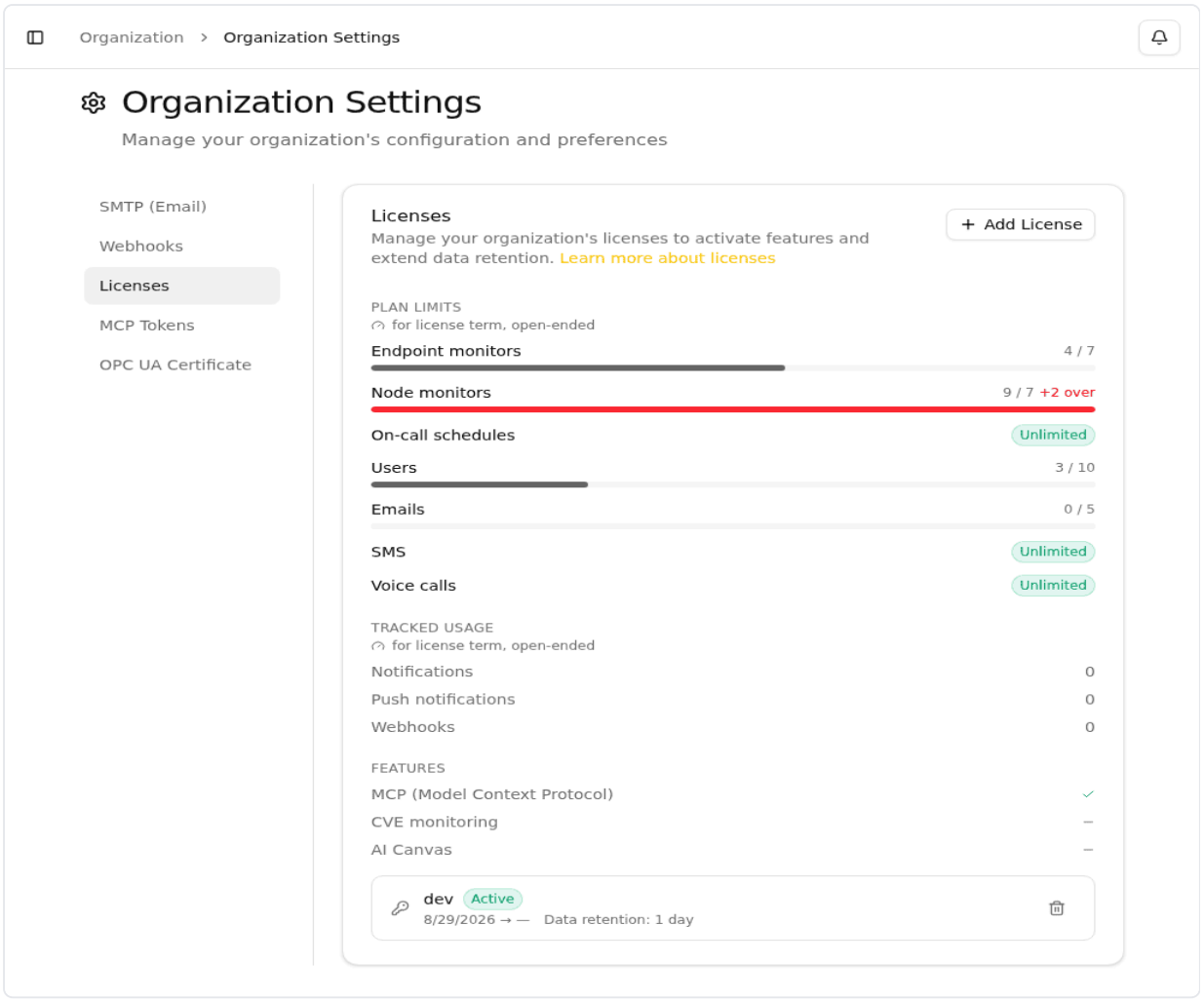
Related

- Members & Roles -- Manage organization members
- Preferences -- Personal notification settings
- OPC UA Configuration -- OPC UA monitors and the application instance certificate

Part 6 · Administration

6.6 Licenses

Licenses activate Pulse features for your organization, define your plan's capacity limits, and control how long monitoring data is retained. Manage them under **Organization > Settings > Licenses**.



Adding a license

1. Navigate to **Organization > Settings > Licenses**.
2. Click **Add License**.
3. Paste your key into the **License Key** field and click **Add License**.

Your license key is provided by Pulse and is a long token that begins with `eyJ`. If the key is rejected, Pulse shows one of these messages:

- **The license key is invalid.** -- The key is malformed or is not a license token.

- **The license key could not be verified.** -- The key's signature did not check out. Confirm you pasted the entire key.

License details

Each license in the list shows:

Field	Description
Instance ID	The Pulse instance the license was issued for.
Status	**Active** (the license currently driving quotas), **Standby** (valid now but not the license in effect), **Expired** (past its end date), or **Pending** (its start date is in the future).
Valid range	Start → end date. A missing date means there is no limit in that direction (open-ended).
Data retention	How many days Pulse keeps your monitoring data before it is automatically deleted.

To remove a license, click the trash icon and confirm.

> **WARNING**
>
> Deleting a license can cost your organization access to licensed features, and the application may no longer be fully usable. Only delete a license you intend to replace.

Usage and plan limits

The usage panel at the top of the Licenses tab compares your current usage against the active license. Each group is labelled with the license term -- **for license term, ends {date}**, or **for license term, open-ended** when the license has no end date.

Plan limits

These capacities are governed by your plan. Each is shown as a gauge of usage against your allowance:

Limit	What it counts
Endpoint monitors	Data sources you have configured.
Node monitors	Data points you have chosen to keep a history for.
On-call schedules	On-call schedules in your organization.
Users	Active members in your organization.
Emails	Email notifications sent during the license term.
SMS	SMS notifications sent during the license term.

A gauge appears in one of three forms:

- **A bar with a count** (for example `12 / 20`) -- your usage against the allowance. The bar is neutral normally, turns **amber** with a *near limit* note at 80% or more, and **red** with a *+N over* note once you exceed the allowance.
- **Unlimited** -- your plan sets no cap on this capacity.
- **not in your plan** -- this capacity is not part of your current plan. Any usage is shown in red as a signal that it falls outside your plan's limits.

Tracked usage

Pulse also reports message volume that is **counted but not capped**, over the same license term:

- **Notifications**
- **Push notifications**
- **Webhooks**

These figures are informational -- they do not block activity when they grow.

Low allowance notifications

When your email or SMS allowance drops to 10% or less of the licensed cap, Pulse automatically sends an in-app notification to every organization admin and owner:

- **Email allowance running low** -- You have used X of Y messages in the current license term. Top up to avoid interrupted delivery.
- **SMS allowance running low** -- You have used X of Y messages in the current license term. Top up to avoid interrupted delivery.

The notification is sent once per license. Installing a new license (for example after a top-up) re-arms the warning for the new term.

License status banner

When your organization's license needs attention, Pulse shows a banner at the top of every page:

- **No valid license** -- "This organization does not have a valid license. Some features may be limited."
- **License expiring soon** -- "Your license is about to expire. Please renew to avoid service disruption."

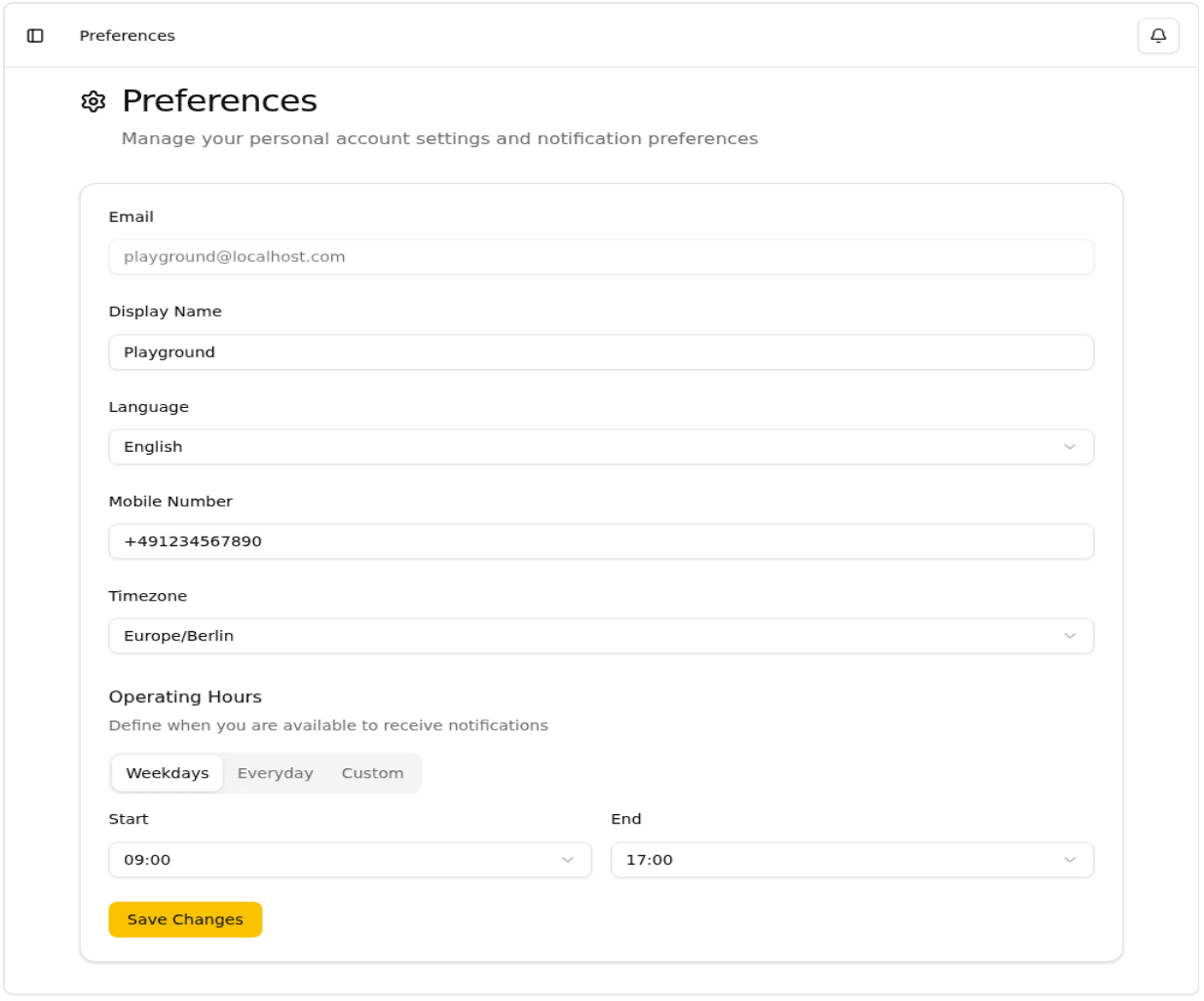
The banner clears automatically once a valid, non-expiring license is active.

Related

- Organization Settings -- SMTP servers and webhooks
- Members & Roles -- Who can administer the organization
- Monitors -- The monitors counted against your plan

6.7 Preferences

The preferences page allows you to customize your personal Pulse settings.



Available Settings

Setting	Description
Display Name	Your name as shown to other team members
Email	Your email address (read-only)
Language	Interface language: English or German
Mobile Number	Phone number for SMS and voice call notifications

Setting	Description
Timezone	Your local timezone for displaying dates and times
Operating Hours	When you are available to receive notifications

WARNING

If any severity in your organization uses SMS or voice call notifications, a warning banner is shown until you add a mobile number. Without a mobile number, those notification channels cannot reach you.

Mobile App Pairing

You can pair the Pulse mobile app to your account without manually entering credentials. This is available from the user menu in the sidebar on desktop.

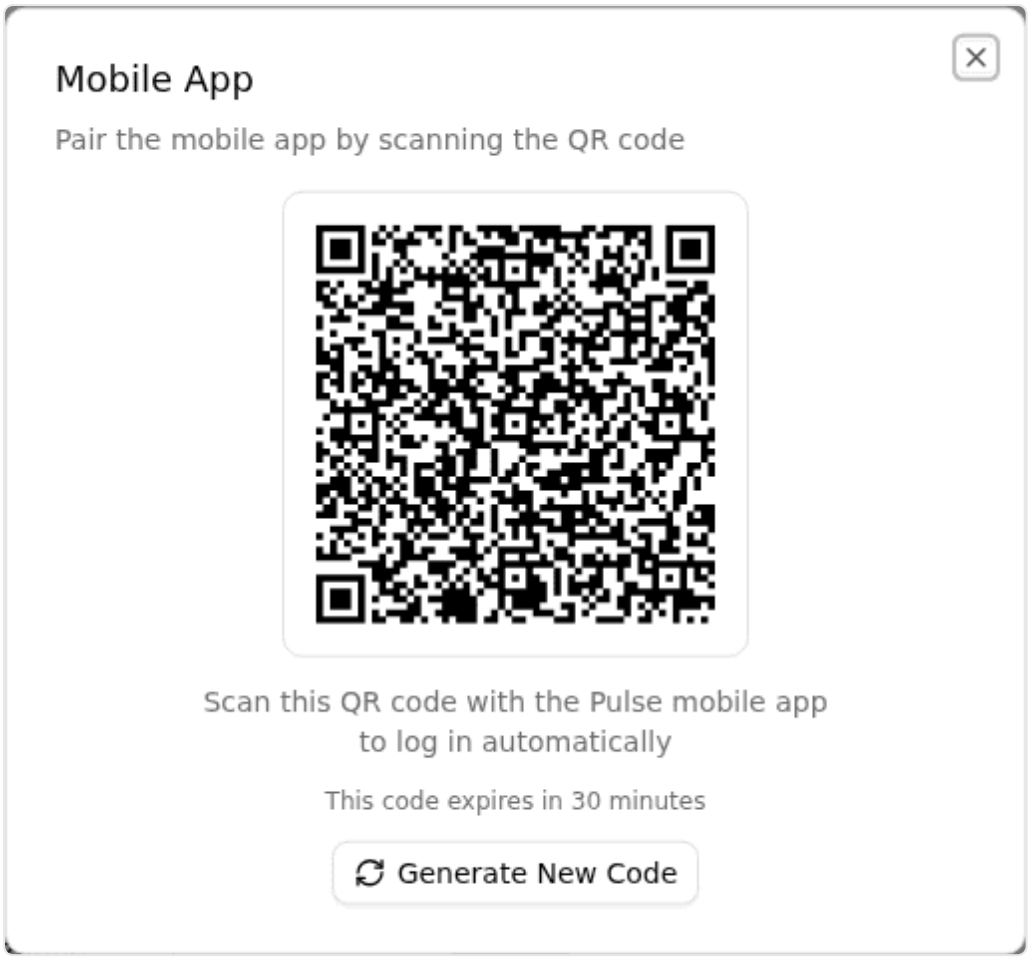
To pair the mobile app:

1. Open the user menu at the bottom of the sidebar.
2. Select **Pair Mobile App**.
3. Scan the displayed QR code with the Pulse mobile app.

The QR code expires after 30 minutes. If it expires before you scan it, click **Generate New Code** to get a fresh one.

Certificate fingerprint

When pairing, the dialog shows a **Certificate fingerprint** — a short sequence of characters derived from the appliance's TLS certificate. Your phone will display the same fingerprint during pairing. Verify that both match before confirming the connection. If they differ, do not proceed.



Operating Hours

Operating hours define when you can be contacted for incident notifications. Outside of operating hours, notifications will be deferred or routed to other team members.

Modes

- **Weekdays** -- Monday through Friday with the same hours each day
- **Everyday** -- Same hours every day including weekends
- **Custom** -- Different hours for each day of the week

Invert operating hours

Enable **Invert operating hours** to flip the window: notifications arrive *outside* the selected hours instead of during them. This is useful if you want to be available nights and weekends but not during regular business hours.

Default Configuration

The default operating hours are Monday through Friday, 09:00 to 17:00 in your configured timezone.

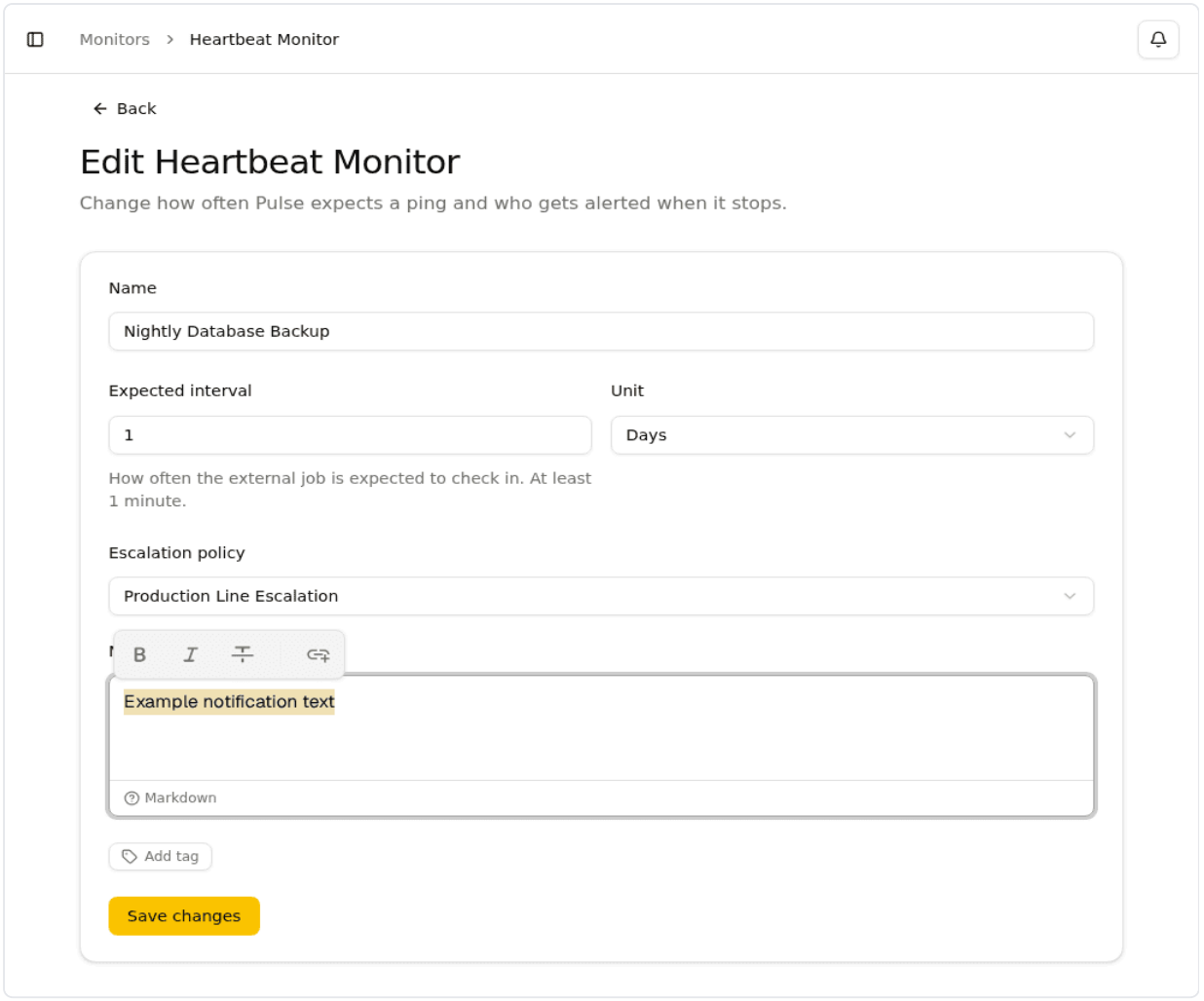
TIP

Configure your operating hours to match your actual working schedule. This ensures you only receive notifications when you can respond to them.

Part 7 · Reference

7.1 Markdown

The message template editor in Pulse supports Markdown formatting. Use the toolbar buttons for quick formatting, or type Markdown syntax directly.



Text Formatting

Syntax	Result
`**bold text**`	**bold text**
`*italic text*`	*italic text*
`` `inline code` ``	`inline code`

TIP

Select text in the editor to reveal the floating toolbar with **Bold**, **Italic**, and **Code** buttons.

Headings

```
# Heading 1

## Heading 2

### Heading 3

#### Heading 4

##### Heading 5
```

Lists

Bullet list:

```
- First item
- Second item
- Third item
```

Numbered list:

```
1. First step
2. Second step
3. Third step
```

Links

```
[Link text](https://example.com)
```

WARNING

Only `https://` and `http://` links are supported.

Blockquotes

```
> This is a blockquote.
```

Code Blocks

````
```
Code block content
```
````

Images

```
![Description](https://example.com/image.png)
```

Paragraphs

Separate paragraphs with a blank line between them.

```
First paragraph.

Second paragraph.
```

Related

- Message Templates

7.2 Troubleshooting

Common issues and their solutions when working with Pulse.

Connection Issues

OPC UA Server Not Reachable

Symptoms: The data source reads "Not reading", and the monitor that alerts when it is unreachable is Unhealthy.

Possible Causes:

- Firewall blocking TCP port 4840 (or custom port)
- OPC UA server not running
- Incorrect endpoint URL

Solutions:

1. Verify the OPC UA server is running on the target machine
2. Check firewall rules between Pulse and the OPC UA server
3. Confirm the endpoint URL is correct, including protocol and port

S7 PLC Not Reachable

Symptoms: The data source reads "Not reading", and the monitor that alerts when it is unreachable is Unhealthy.

Possible Causes:

- Network connectivity issue
- Incorrect IP address, rack, or slot configuration
- PLC is in STOP mode

Solutions:

1. Verify network connectivity to the PLC (ping the IP address)
2. Check rack and slot values match your PLC hardware
3. Verify the PLC is in RUN mode

Authentication Issues

OPC UA Authentication Failed

Symptoms: The data source cannot connect and reports an authentication error.

Solutions:

1. Verify username and password are correct
2. Check that the security mode and policy match the server configuration
3. For certificate authentication, ensure the certificate is valid and trusted by the server

Notification Issues

Not Receiving Notifications

Possible Causes:

- Operating hours configured incorrectly
- Not assigned to the escalation policy
- Notification channel not configured in severity

Solutions:

1. Check your operating hours in Preferences
2. Verify you or your team are listed as recipients in the escalation policy
3. Confirm the severity level includes the expected notification channels
4. Verify your mobile number is set correctly for SMS/call notifications

7.3 SBOM

Every shipped Pulse build includes a Software Bill of Materials (SBOM): a machine-readable inventory of the third-party components bundled in the product.

Pulse publishes its SBOM in the CycloneDX JSON format. It is a single merged document covering all shipped components:

- The Haskell backend (resolved from the pinned dependency set)
- The Go services — the OPC UA collector, the S7 collector, and the appliance CLI
- The web and mobile front-ends (npm dependencies)

Download

Download the Pulse SBOM (CycloneDX JSON)

You can also open it in your browser.

Using the SBOM

The CycloneDX file can be ingested by common supply-chain and vulnerability tooling — for example Dependency-Track, Grype, or Trivy — to audit licenses and scan the components Pulse ships for known vulnerabilities.

Part 8 · Mobile App

8.1 Overview

The Pulse mobile app is a companion to the Pulse web application. Once it is connected to your Pulse server, you can monitor your OT infrastructure and receive alert notifications on the go.

Availability

The mobile app is currently in pre-release and is not yet generally available on the App Store or Google Play. Contact your Pulse administrator if you would like access.

Supported platforms

- **iOS** — iPhone and iPad
- **Android** — *coming soon*. The Android version is in development and not yet available for installation.

The interface adapts automatically to your device's light or dark appearance setting.

What you can do

The mobile app gives you the same Pulse experience you have in the browser. From your phone you can:

- View the Dashboard with live status of all your monitors
- Browse and inspect Monitors
- Track and acknowledge Incidents
- Open the Notification Center to review recent alerts

Authentication, organization switching, and team management all happen inside the same Pulse interface you already know — there is no separate mobile login.

What is unique to the mobile app

Three things are handled by the mobile shell rather than the web application:

- Server pairing — point the app at your Pulse server on first launch and change it later if needed.
- Push notifications — receive alerts on your device, driven by your existing on-call rotation and escalation policies.

Related

- Setup & Pairing — connect the app to your Pulse server
- Notifications & Settings — enable alerts and change the server address later
- Dashboard — overview of all monitors
- Incidents — investigate and acknowledge active incidents

8.2 Setup & Pairing

Before you can use the Pulse mobile app, you need to point it at your Pulse server.

First-run pairing

When you open the app for the first time, the **Connect to Pulse** screen asks for the address of your Pulse server.

1. Enter the server address — for example, `pulse.yourcompany.com`.
2. Tap **Connect**. The app verifies that the server is reachable before continuing.
3. If the check succeeds, the app loads your Pulse server and takes you straight to the sign-in screen.

> **TIP**
>
> You can omit `https://` — the app will add it for you.

If you do not have your own server address yet, tap **Use Pulse Cloud (app.pulsehq.de)** to point the app at the hosted Pulse Cloud service.

If the address cannot be reached, the app shows an error message and offers a **Use this address anyway** option. Use it when you know the server is reachable from your network even though the initial check failed (for example, on internal networks that block the verification request).

Signing in

Once the app is paired with a server, sign-in works exactly like the Pulse web application. Use the same email address and password you would use in your browser. Switching between organizations, managing your profile, and joining teams all happen inside the same interface — see Members & Roles and Teams.

Changing the server address later

You can change the server address at any time from inside the app:

1. On the home screen, **press and hold the small handle at the very top of the screen** for about 1.5 seconds. The **Server Configuration** sheet slides up.
2. Edit the server address, then tap **Connect** to verify and save it.
3. To return to the hosted service, tap **Reset to Default**.
4. To pair with a completely different server from scratch, tap **Disconnect and reconfigure**. This takes you back to the first-run pairing screen.

When the device is offline

If your phone loses network connectivity, or your Pulse server is briefly unreachable, the app shows a **Connection lost** screen instead of the main interface.

- The app retries the connection automatically with a short delay between attempts.
- Tap **Try again** to retry immediately.
- Tap **Server configuration** to open the same settings sheet described above — useful if the server address itself needs to change.
- Expand **Details** to see exactly which server the app is trying to reach and the last error it encountered (for example, an HTTP status code or a network timeout).

The app reconnects automatically as soon as the network returns or the server becomes reachable again — you do not need to take any action.

> **TIP**
>
> Bringing the app back to the foreground (for example, after switching apps) also triggers an immediate reconnect attempt.

Verifying the appliance's fingerprint

If you are pairing with a self-hosted Pulse appliance rather than Pulse Cloud, the app may ask you to confirm a SHA256 fingerprint after the initial connection. Compare the value on screen with the one displayed in the appliance's TLS Certificates screen — they must match character-for-character before you continue.

Related

- Notifications & Settings — enable push notifications and access the settings sheet
- TLS Certificates — the operator-side screen that shows the fingerprint used during pairing

- Troubleshooting — general help with sign-in and connectivity issues

Part 8 · Mobile App

8.3 Notifications & Settings

The mobile app can deliver alert notifications directly to your device, so you do not need to keep the Pulse web application open to know when something needs your attention.

Granting notification permission

The first time the app loads your Pulse server, your operating system asks whether Pulse may send you notifications. Tap **Allow** to enable push notifications on this device.

If you decline the prompt, the app continues to work but no push notifications are delivered. You can change your decision later in your device settings — see Re-enabling notifications below.

What you'll receive

Push notifications follow the same rules as every other Pulse alert channel. Whether your device is paged for a given incident depends on:

- The severity of the incident
- The escalation policy that applies to the affected monitor
- Who is currently on the on-call schedule

To choose which channels (push, email, SMS, voice) you want to receive for each severity, open the Notification Center or your account Preferences — these settings are shared between the web application and the mobile app.

Reading the notification icons

Every push notification starts with an icon that tells you what happened before you open it — useful when several notifications for the same incident are stacked on the lock screen:

Title	What it means
□ *Incident name*	A new incident was raised and you are being paged for it.
□ *Incident name* acknowledged	Someone took the incident on. The body names who — you do not need to respond as well.

Title	What it means
Incident name resolved	The incident is closed. Nothing is left to do.
Incident name	A colleague escalated this incident to you personally, outside the automatic escalation policy. The body says who sent it.

A page is delivered even when the incident is already acknowledged and even outside your configured operating hours, because somebody chose you deliberately — see Incidents for how escalating works.

The icons appear in push notifications on your device. The in-app Notification Center in the web application lists a different set of messages and does not use them.

Re-enabling notifications

If you previously declined the notification prompt, or you want to re-enable notifications later, do it from your device settings:

- **iOS** — open **Settings**, scroll down to **Pulse Connect**, tap **Notifications**, and enable **Allow Notifications**.

Return to the Pulse app afterwards; the new permission state is picked up automatically.

Settings sheet

The mobile shell only has a few settings of its own. They are accessible from the **Server Configuration** sheet:

1. On the home screen, press and hold the small handle at the very top of the screen for about 1.5 seconds.
2. The sheet slides up from the bottom and offers:
   - **Server URL** — change the Pulse server this device is paired with.
   - **Connect** — verify and save the new address.
   - **Reset to Default** — return to the hosted Pulse Cloud service.
   - **Disconnect and reconfigure** — clear the stored address and go back to the first-run pairing screen.

For a full walk-through of pairing and re-pairing the app, see Setup & Pairing.

Related

- Notifications — the in-app Notification Center
- Severities — choose which channels are used for each severity
- Preferences — set up email, SMS, and voice channels
- Setup & Pairing — connect the app to your Pulse server

Part 9 · Appliance

9.1 Overview

The Pulse appliance is a turnkey NixOS device that runs the Pulse server out of the box. Plant operators and on-site IT staff manage it through a terminal user interface (TUI) on the console — there is no need to log in to a shell or edit configuration files by hand.

How to reach the TUI

- **On the console**: the appliance auto-logs in as the `pulse` user on tty1 and launches the TUI directly. Plug in a monitor and keyboard to use it locally.
- **Over the network**: SSH into the appliance as the `pulse` user. The TUI starts automatically on login.

In this section

- Installation — install the appliance from the ISO image and complete first-boot setup.
- Networking — configure the appliance's IP address, gateway, and DNS.
- TUI Reference — every menu item the operator sees on the console, in one place.
- TLS Certificates — manage the certificate the web interface presents to browsers.
- SFTP Access — onboard customer SFTP keys for file exchange.
- SSH Access — grant support engineers shell access to the appliance.
- System Updates — apply a signed update bundle from USB, CD/DVD-ROM, or SFTP, with automatic backup and rollback.
- Operations — take database backups and roll back to a previous version.

9.2 Installation

This page walks through installing the Pulse appliance onto a fresh machine. The procedure is the same whether the target is a small industrial PC, a rack server, or a virtual machine — boot the ISO, answer a handful of questions in the installer TUI, and wait for the appliance to reboot into the operator interface.

The steps below match the order of the installer wizard you see on the console, one-for-one.

Hardware requirements

The appliance is intentionally modest in what it needs:

- **CPU**: 2 cores (x86_64). Any modern Intel or AMD CPU works.
- **RAM**: 4 GB minimum, 8 GB recommended for sites with more than a few hundred monitors.
- **Disk**: 32 GB minimum. SSD strongly recommended — the database does enough random I/O that a spinning disk noticeably slows the web UI.
- **Firmware**: UEFI or legacy BIOS — both are supported. The ISO is a hybrid image that boots either way.
- **Network**: one ethernet port. Wi-Fi is not supported.

The target disk will be **completely erased** during installation. If the machine has more than one disk, the installer lets you pick which one to use, but everything on the selected disk is gone once you confirm.

Flashing the ISO to a USB drive

You need a USB drive of 4 GB or more. Any tool that does a raw block copy of the ISO onto the drive will work — the image is bootable as-is, with no partitioning or formatting step required.

- **Linux / macOS**: `sudo dd if=pulse.iso of=/dev/sdX bs=4M status=progress conv=fsync` (replace `/dev/sdX` with the actual device — `lsblk` or `diskutil list` will identify it). Get the device wrong and you wipe the wrong disk; double-check before pressing `enter`.
- **Windows**: use Rufus in **DD image** mode (not ISO mode). Select the ISO, select the USB drive, click start.

- **Cross-platform GUI**: Balena Etcher works on all three operating systems and hides the manual device-picking.

The resulting USB drive boots on both UEFI and legacy-BIOS machines, so the same stick can install any target you have on hand.

First boot

Plug the USB drive into the target machine and power it on. You may need to press a key during the firmware splash to open the boot menu — typical keys are `F12`, `F11`, `F10`, `F2`, `Esc`, or `Del`, depending on the vendor.

In the boot menu, pick the USB drive. The appliance ISO loads and, after a few seconds, drops you into the installer TUI on tty1. There is no graphical installer — the entire setup is keyboard-driven from the console.

If the machine boots back into its existing OS instead of the USB drive, the boot order in firmware is wrong. Reboot, enter firmware setup, and either change the boot priority or use the one-time boot menu again.

Installer wizard

The wizard has eight steps in fixed order. Navigate with the arrow keys (or `j` / `k`), `enter` to confirm a step, `esc` to step back. `ctrl+c` aborts the installation entirely.

Welcome

The opening screen names the wizard and summarises what comes next: disk selection, network configuration, optional NTP, and base URL. It also reminds you that the target disk will be erased. Press `enter` to begin or `ctrl+c` to back out.

Disk

The installer scans the machine and lists every disk it could install onto. Each row shows the device name (for example `/dev/sda`), its size, and the model string the firmware reports. Use the arrow keys to highlight the target and press `enter` to select it.

Warning. The disk you select here is wiped completely. There is no undo once the installation step starts. If the machine has both a working OS disk and a separate empty disk, take a moment to confirm which is which — vendor model strings and sizes are usually enough to tell them apart.

Network

Pick how the appliance gets its IP address:

- **DHCP** — the network already hands out addresses automatically. Most internal LANs and many plant networks work this way. Choose this if you do not know.
- **Static IP** — the LAN team has reserved a fixed IP for the appliance. Choosing this reveals three extra fields: the IP address (in CIDR notation, e.g. `192.168.1.100/24`), the gateway, and a DNS server.

The static-IP fields are validated before the wizard moves on — typos or impossible values produce an error and you stay on the page until they are fixed. See Networking for guidance on choosing static-IP values and switching modes later.

NTP

Optional. Leave the field empty to use the default upstream NTP pool — fine when the appliance has internet access. If the site uses an internal time server, type its hostname or IP here and the appliance will sync against that instead. Press `enter` to continue.

BaseURL

The URL that the web UI and the mobile pairing flow will use to reach the appliance. This is the address operators type into a browser and that the mobile app stores when a phone is paired.

- If the appliance has a DNS name on the LAN, use that (`https://pulse-system.example.local`).
- If not, use the IP you assigned in the previous step (`https://192.168.1.100`).
- Always include the scheme (`https://`). Plain hostnames are rejected.

A bad base URL can be changed later from the TUI, but mobile pairings created against the old URL will need to be redone — so it pays to get this right the first time.

Confirm

The wizard shows everything you entered on one screen: disk, network mode (and static-IP details, if any), NTP server, and base URL. It repeats the warning that the disk will be erased. Press `enter` to start the actual installation, or `esc` to back up and change something.

Progress

After confirmation, the installer partitions the disk, copies the system image, and writes the configuration. Progress lines stream to the screen as each step completes. Press `l` at any time to open the full installer log — useful when the wizard pauses on a slow

operation and you want to see what is actually happening.

The installation typically takes a few minutes on an SSD; spinning disks can take noticeably longer because of the partitioning and copy steps. Do not power off the machine during this phase.

Done

When the installer finishes, you land on the **Done** screen. On success it confirms the installation completed and offers a reboot — press `enter` to reboot, and remove the USB drive while the BIOS POST runs so the machine boots from the freshly installed disk.

If the install failed, the screen shows the error instead; `l` opens the log so you can see what went wrong, and `esc` returns you to the Confirm step so you can adjust an answer and retry. `ctrl+c` exits without rebooting.

After the reboot the appliance comes up with the TUI as the default console session, ready for day-to-day operation.

9.3 Networking

This page covers day-2 networking changes — how to swap an appliance between DHCP and a fixed IP, what the static-IP fields mean, and how the apply / revert flow protects you from locking yourself out. The screen described here is **Network Configuration** in the operator TUI.

The networking mode is also chosen during installation, but on a running appliance you change it from this screen rather than reinstalling.

DHCP vs static — which to pick

The Network Configuration screen offers two modes at the top: **DHCP (automatic)** and **Static IP**. The screen always starts in whatever mode the appliance is currently using; move the highlight up or down to switch.

- **DHCP** is the default and is the right choice when the network already hands out addresses automatically. Most office LANs and many plant networks fit this pattern. You do not need to type anything else — DHCP mode has no extra fields.
- **Static IP** is for environments without a DHCP server, or when the appliance needs a fixed address. Two situations make static unavoidable: the mobile pairing flow stores the appliance's address in each phone, so a DHCP lease that changes silently breaks every paired device; and any external monitoring or reverse proxy that points at the appliance needs an address that does not move.

Switching modes is non-destructive — the previous configuration is snapshotted before anything is written, so you can flip from DHCP to static (or back) and revert if the new setting does not work. See Apply, revert, and backups.

The three static-IP fields

When Static IP is selected, three fields appear below the mode picker. Use `tab` (or the arrow keys) to move between them.

- **IP Address** — the appliance's address, written in CIDR notation: address plus subnet-mask length, separated by a slash. Example: `192.168.1.100/24`. The `/24` part is required — without a prefix length the appliance has no way to know which other hosts are on its subnet.

- **Gateway** — the IP address of the router that forwards traffic off the local subnet. Without this, the appliance can talk to other hosts on the same LAN but cannot reach the internet or any other network. Ask your LAN team for the gateway address if you do not already have it; it is usually the first or last usable address in the subnet (for example `192.168.1.1` or `192.168.1.254`).
- **DNS Server** — the IP address of the resolver the appliance should use to translate hostnames into IP addresses. Internal DNS at the customer site is usually preferred when one exists, because it can resolve site-local names (file servers, plant systems) that public resolvers cannot. A public resolver like `8.8.8.8` works as a fallback when no internal resolver is available.

The TUI refuses obviously malformed input: an IP without a prefix length, or a gateway or DNS server that is not a valid IPv4 address (hostnames like `dns.example.com` are not accepted — use the resolver's IP). The error appears in red below the fields and the change is not applied — fix the value and press apply again.

Apply, revert, and backups

The bottom of the screen shows the available keystrokes:

```
tab navigate • ctrl+s apply • ctrl+r revert • esc back
```

`Ctrl-S` **applies the change.** Before writing the new configuration, the appliance takes a snapshot of the previous one. If everything looks good — gateway pings, DNS resolves, the web UI still loads — you are done. A short confirmation (`Network configuration applied`) appears under the form on success.

`Ctrl-R` **reverts to the previous configuration.** This is the safety net for the case where you locked yourself out. A typical scenario: an operator switches to static IP, types the wrong gateway, and the appliance is no longer reachable from the workstation. The console TUI is still available locally — log in there, open Network Configuration, press `Ctrl-R`, and the previous (working) settings come back. The screen confirms the rollback with `Reverted to previous network configuration`.

Below the message area the screen prints how many previous configurations are kept for rollback (for example `2 previous config(s) available for rollback`). The count climbs each time you apply a change, so a quick test of "did my change actually do anything" is to apply, see the count tick up, and confirm.

Press `Esc` to leave the screen without applying. Edits in the input fields are discarded — nothing is written until `Ctrl-S` is pressed.

A safe sequence for a static-IP change

When you do not have physical console access and are working over SSH, the revert path is the difference between a five-minute change and a site visit. The sequence we recommend:

1. Make sure you have a way to reach the console — either physically at the appliance, or via remote KVM/iLO/iDRAC if the hardware supports it. The TUI on the console always works regardless of the network configuration.
2. Note the current settings on a piece of paper before you start, so you know what "working" looked like.
3. Switch to static, fill in the three fields, and press `Ctrl-S`.
4. From a second workstation (not the one you are SSH'd in from), confirm the appliance is reachable on the new IP — pinging it, loading the web UI, or both.
5. If anything looks wrong, log in on the console and press `Ctrl-R`. The previous configuration is restored and the appliance is reachable again.

Where the configuration lives

> **Managed by the TUI — do not edit by hand**
>
> The networking settings are persisted in `/var/pulse/site.env`. Treat that file as TUI-managed: every change should go through the Network Configuration screen so the backup snapshot is taken and the dependent services (networkd, the web UI base URL) are reloaded in the right order. Editing the file directly bypasses both, and the next apply from the TUI will overwrite your changes anyway.

If you suspect the file is corrupt or out of sync, the safest recovery is the same revert path described above — `Ctrl-R` rolls back to the previous good snapshot rather than asking you to hand-edit the file.

Related screens

The Network Configuration screen handles IP-level settings only. Two adjacent concerns live elsewhere in the TUI:

- **NTP Configuration** — sets the time-synchronization server. Time and IP are independent: changing the network mode does not change the NTP setting, and vice versa.

- **TLS Certificates** — manages the certificate the web UI presents. The certificate's Subject Alternative Names (SANs) include the appliance's IP, so after a static-IP change you may want to regenerate or reinstall the certificate to match the new address.

See the TUI Reference for the full list of menu entries and the Installation page for the equivalent network step during first-boot setup.

9.4 TLS Certificates

This page covers the operator's side of managing the TLS certificate that the appliance presents on its web interface. The screen described here is **TLS Certificates** in the operator TUI.

The screen has two parts.

- The **status panel** at the top shows the subject, issuer, validity window (with an in-line warning when the certificate is within 30 days of expiry), the IP and DNS SANs the certificate covers, the SHA256 fingerprint, and a `Type:` line that says whether the active certificate is self-signed or customer-supplied.
- The **actions list** at the bottom offers **Generate CSR**, **Upload Certificate**, and **Restore Self-Signed** — plus **Cancel CSR** when a request is awaiting signature.

If the SFTP exchange directory already contains staged certificate files when the operator opens the screen, a hint such as `1 cert ready in SFTP — Upload to apply` appears under the status panel as a reminder that there is work to do.

Press `r` at any time on the main view to reload the panel — useful after a customer has just uploaded files over SFTP and you want the hint to appear without leaving the screen.

What ships out of the box

A freshly installed appliance generates a **self-signed** certificate on first boot and serves it on the web interface. There is no public CA signature involved, so browsers will warn on the first visit — that is expected.

For headless or LAN-only deployments the self-signed certificate is often sufficient: the mobile app does not rely on a public CA at all. It verifies the appliance by **fingerprint** instead, which makes pairing safe even when the certificate is self-signed. See Fingerprint and mobile pairing below for the verification flow.

If you do want browsers to trust the certificate without a warning — or you have policy reasons to install a customer-supplied certificate — use **Upload Certificate** as described next. To go back to the appliance-generated certificate at any time, use **Restore Self-Signed**.

Requesting a CA-signed certificate

If your CA signs certificates from a certificate signing request (CSR), the appliance can generate the request and the matching private key for you — the private key never leaves the appliance.

1. From the **TLS Certificates** screen, choose **Generate CSR** and confirm. The active certificate keeps serving traffic; only a new pending key is created. The request reuses the appliance's IP and DNS names, so the signed certificate covers the same addresses.
2. Connect over SFTP and download `pulse-csr.pem` from the `downloads/` folder, then submit it to your CA. SFTP access is set up on the SFTP Access screen.
3. While the request is awaiting signature, the status panel shows a `CSR pending CA signature` reminder and a **Cancel CSR** action appears.
4. When the CA returns the signed certificate, install it with **Upload Certificate** (see below). Because a request is pending, the appliance matches the certificate against the pending key automatically — you do not upload a key.

If a request already exists and you generate another, the pending key is replaced and any CSR you already submitted becomes invalid. To abandon a request without installing anything, choose **Cancel CSR** — it removes the pending key and request and leaves the active certificate untouched.

Uploading your own certificate

The **Upload Certificate** action accepts a PEM-encoded certificate and, optionally, a PEM-encoded private key. Two delivery paths are supported: a USB drive plugged into the appliance, or files staged into the SFTP exchange directory. The validation, install, and reload steps are identical in both cases — only the source picker differs.

Both paths accept only PEM-encoded inputs. If a file is not parseable as PEM, or if the certificate and key do not match, the TUI reports the error inline and **does not install** anything; the previously active certificate continues to serve traffic.

The key prompt also adapts to whether a CSR was generated earlier:

- If there is a pending key in `tls-key-pending.pem` (the operator generated a CSR and is now installing the signed certificate that came back from the CA), uploading the certificate alone matches it against the pending key.
- Otherwise the certificate is matched against the currently active key — useful when only the certificate itself is being rotated.

- If the operator picks a key file too, the cert/key pair is validated together before anything is written to disk.

Via USB

Use this path when you are at the appliance with a thumb drive in hand.

1. Copy the PEM certificate (and the private key, if you are rotating it) to a USB drive formatted as FAT32 or ext4. Files must have the `.pem` extension to be discovered by the picker.
2. Plug the drive into the appliance. The TUI mounts it read-only at `/mnt/usb`.
3. From the **TLS Certificates** screen, choose **Upload Certificate**, then **USB** as the source.
4. If more than one USB drive is detected, pick the right one. The TUI then scans for `.pem` files and shows a picker. If no `.pem` files are found, the screen returns to the source picker with an error — copy the files to the drive's root or a top-level directory and try again.
5. Select the certificate file and press `Enter`. The TUI parses the PEM. If parsing fails it shows a red error and stays on the picker — fix the file and try again.
6. The TUI then asks whether you also have a new key file to upload. Pick the key from the same drive, or skip if you are reusing the currently active key (or, if a CSR was generated previously, the pending key in `tls-key-pending.pem`).
7. On success the certificate is installed, nginx is reloaded with the new pair, and the screen returns to the main view with a green confirmation. The drive is unmounted automatically.

If you press `Esc` partway through the flow — for example after picking the wrong cert file — the TUI walks back through the previous step. From the cert picker, `Esc` returns to the source picker and unmounts the drive; from the key prompt, `Esc` returns to the cert picker so you can choose again.

Via SFTP

Use this path when the customer wants to deliver the certificate over the network rather than carry a USB stick. The prerequisite is that the customer can authenticate to the SFTP exchange — by SSH key or the appliance's password; see SFTP Access to set that up.

1. The customer uploads the certificate (and key, if separate) into the exchange directory under `uploads/certificates/`. From their perspective they SFTP to the appliance as the `pulse-transfer` user and `put` the files.
2. The TLS Certificates screen will show a hint such as `1 cert ready in SFTP — Upload to apply` when files are waiting. Choose **Upload Certificate**, then **From SFTP exchange** as the source.

3. The TUI lists every `.pem` file staged under `uploads/certificates/`. Pick the certificate, answer the key prompt, and confirm. Validation rules are the same as the USB path.
4. On success the consumed files are removed from the exchange directory and an audit entry is written so the operator can later show which customer-supplied file was applied and when.

If a file in the exchange directory is not a valid PEM, the TUI reports it and skips installation — the file remains in `uploads/certificates/` for the customer to replace. The same applies to certificate/key mismatches: nothing is installed, and the staged files are kept so the customer can correct and re-upload.

If you reach this screen and the SFTP list is empty, the most common cause is that the customer uploaded into the wrong subdirectory. Files for this picker must land specifically under `uploads/certificates/`; bundles or unrelated PEM files elsewhere in the exchange are ignored by the TLS Certificates screen.

Restoring the self-signed certificate

The **Restore Self-Signed** action regenerates an appliance-controlled self-signed certificate and switches the web interface back to it. Reach for this when:

- The uploaded customer certificate turns out to be wrong (mismatched subject, expired before installation, signed by the wrong CA).
- The uploaded certificate has expired and no replacement is ready — restoring the self-signed certificate returns the appliance to a working HTTPS state so operators can still reach the web UI.
- You are wiping a customer-specific configuration off the appliance ahead of redeploying it elsewhere.

The confirmation prompt warns that the current certificate will be replaced and that nginx will reload. Any pending CSR is cleaned up as part of the restore — a stale `tls-key-pending.pem` left over from a half-finished CSR cycle is best-effort removed. If the current certificate is already self-signed, the prompt offers to regenerate it anyway, which produces a fresh fingerprint while keeping the same `Type: Self-signed` label.

The new self-signed certificate is regenerated using the appliance's current site IP (read from `/var/pulse/site.env`) as one of the SANs. In DHCP mode the site IP may be empty, in which case the IP SAN is omitted; this is normal and does not affect the mobile pairing flow, which relies on the fingerprint rather than the SAN.

Fingerprint and mobile pairing

The **Fingerprint** row on the main TLS Certificates panel shows the SHA256 fingerprint of the active certificate, formatted with colon separators (for example, `AB:CD:EF:...`). This is the value the Pulse mobile app asks the operator to verify during first-run pairing.

Because the appliance often serves a self-signed certificate, the mobile app cannot rely on a public CA to authenticate the server. Instead, the app shows the fingerprint of the certificate it received and asks the operator to compare it with the value displayed on this screen. They must match character-for-character. If they do not match, abort pairing — the connection is being intercepted, or the operator is looking at the wrong appliance.

The fingerprint changes whenever the certificate changes — including every time **Restore Self-Signed** is used, since that regenerates the key pair. After restoring (or uploading a new certificate) any already-paired mobile devices will surface a fingerprint-mismatch warning the next time they connect; reconfirm the new fingerprint from this screen.

Related screens

- SFTP Access — authorise the customer, by SSH key or the appliance password, for the SFTP upload path described above.
- Mobile Setup & Pairing — how the mobile app uses the fingerprint shown on this screen to verify the appliance.
- TUI Reference — the full operator menu.

9.5 SFTP Access

This page covers the operator's side of authorising someone outside the appliance — a vendor, a customer admin, or an on-call support engineer — to drop files into the appliance over SFTP. The screen described here is **SFTP Access** in the operator TUI.

Access here authorises file *delivery only*, whether the remote party connects with their own SSH key or with the appliance's SFTP password. Neither grants a shell on the appliance — that is a separate concern handled by the SSH Access screen.

When you need this

SFTP access is the answer when a file needs to land on the appliance and a USB stick is not practical. Two recurring situations:

- **A vendor needs to deliver an update bundle remotely.** Pulse support has built a new system tarball and signed it, and the operator does not want to drive to the plant to plug in a USB drive. Authorising the vendor's public key here lets them SFTP the bundle into the exchange directory; the operator then picks it up from the Update System screen.
- **A customer wants to push their own TLS certificate.** The customer's CA has issued a fresh certificate and key pair for the appliance's web interface, and the customer would rather upload it over the network than walk a USB drive to the rack. Authorising their key here lets them SFTP the PEM files into the exchange directory; the operator installs them from the TLS Certificates screen.

In both cases the authorised party never gets a shell — they can only put files into the exchange directory. The TUI is the boundary that decides what to do with those files.

Password login

Not every remote party has an SSH key, and setting one up just for a single delivery is often more effort than it is worth. For those cases the appliance also accepts a password, which works with any graphical SFTP client — FileZilla, WinSCP, Cyberduck, and the like.

The appliance generates its own SFTP password on first boot and shows it at the top of the **SFTP Access** screen, under **Password login (any SFTP client)**:

- **Host** — the appliance's IP address.
- **Port** — `22`.

- **Username** — `pulse-transfer`.
- **Password** — shown as five groups of four characters (`xxxx-xxxx-xxxx-xxxx-xxxx`), readable straight off the console.

Pass these four values to the remote party over a channel you trust and they can connect right away — no key exchange needed. The session is restricted exactly as it is for key access: chrooted to the exchange directory, with no shell and no forwarding.

There is one password per appliance, shared by everyone who logs in this way — that is the trade-off against keys. A key identifies one party and can be revoked on its own; the password is a single shared secret. Prefer keys when you need to authorise several parties independently, and reach for the password when you just need one delivery to happen quickly.

Regenerating the password

Press `g` to generate a fresh password. If one already exists the TUI asks you to confirm first, because **the current password stops working the moment a new one is generated** — every client using it, including your own, must be updated with the new value. On success the screen returns to the list with `SFTP password regenerated.` and the new password on display.

Regenerate once a delivery is finished, or any time you suspect the password has spread further than intended.

Adding a key

The screen lists the currently authorised keys below the password panel, with the keystroke row at the bottom showing what is available:

```
a add key (paste) • u add from USB • d remove • g regenerate password • esc back
```

There are two ways to add a key. Pick whichever fits the workflow — the result is identical.

Paste the key directly

Press `a`. A text field appears with the placeholder `ssh-ed25519 AAAA... user@host`.

1. Ask the remote party for their SSH public key. It is the contents of a `.pub` file — a single line that starts with the algorithm name (`ssh-ed25519`, `ssh-rsa`, `ecdsa-sha2-nistp256`), then the base64-encoded key material, then an optional comment such as `alice@vendor.com`.

2. Paste the whole line into the field. The terminal must support paste — most modern terminal emulators do, and on the appliance console you typically paste with `Ctrl-Shift-V` or by middle-clicking, depending on the terminal.
3. Press `Enter` to confirm. The TUI validates the line with `ssh-keygen`. If the key is malformed, a red error appears below the field and nothing is added — fix it and try again. If the key is already on the list, the TUI refuses with the existing fingerprint so you know which entry to look at.
4. On success the screen returns to the list, the new key appears with its SHA256 fingerprint and comment, and a green status line confirms `Key added: SHA256:...`.

Press `Esc` to cancel without adding anything.

Pick a `.pub` file from a USB drive

Press `u`. The TUI scans the mounted USB drive for files ending in `.pub` and shows a picker.

1. Plug a USB stick that contains the remote party's `.pub` file into the appliance. The TUI mounts the drive automatically at `/mnt/usb`.
2. Back on the SFTP Access screen, press `u`. If no `.pub` files are found, the screen returns to the list with a red `no .pub files found on USB drive` message — copy the file to the USB drive's root and try again.
3. Move the highlight with the arrow keys (or `j` / `k`) to the right file and press `Enter`. The TUI reads the file, validates it, and adds the key.
4. As with paste, the screen returns to the list on success with a green confirmation and the new fingerprint.

Press `Esc` at any point to back out without adding a key.

The USB picker is the right path when the customer has handed you a thumb drive in person, or when the key was emailed as an attachment and saved to a stick rather than copied to the clipboard.

Listing and removing keys

The main view lists every authorised key, one per line. Each entry shows the SHA256 fingerprint and the key's comment field — usually an email address or a host identifier like `alice@vendor.com`. Use the fingerprint to identify which key is which when several look similar; the comment alone is not authoritative because anyone can put any text there.

To remove a key:

1. Move the highlight with `↑` / `↓` (or `k` / `j`) until the row you want to remove is selected.
2. Press `d`. A confirmation prompt appears showing the fingerprint and comment of the key about to be removed.
3. Press `y` to confirm or `n` (or `Esc`) to cancel.

On confirmation the key is rewritten out of the appliance's `authorized_keys` file and the screen returns to the list with `Key removed`. The removal takes effect immediately — any open SFTP session the holder of that key already has remains connected until they disconnect, but they cannot start a new one.

What the authorised party can do

Both authentication methods — an authorised key or the password — reach a single SFTP user: the `pulse-transfer` principal. That user is chrooted to the appliance's exchange directory, has no shell, and `AllowTcpForwarding`, `X11Forwarding`, and `PermitTunnel` are all off. In effect anyone who authenticates can:

- Connect over SFTP (port 22, username `pulse-transfer`).
- Put files into the exchange directory's `uploads/` subtree.
- Get files from the `downloads/` subtree (used for files the appliance stages for download, such as backup archives and a generated CSR).
- Not log in to an interactive shell.
- Not start a TCP forward, X11 session, or tunnel.

Operator-level shell access — for support engineers troubleshooting the appliance itself — is managed separately on the SSH Access screen. The two screens are independent: authorising a key here does not grant shell access, and authorising one there does not grant SFTP upload rights.

Where the files land

Files uploaded via SFTP land in `/var/pulse-exchange/uploads/`, organised into subdirectories by kind: `updates/` for system update bundles and `certificates/` for TLS PEM files. The exact paths are part of the SFTP user's chroot view — the remote party sees them at the root of their session.

The operator does not consume those files from this screen. Instead:

- **Update System** offers an **SFTP Upload** source that lists the bundles staged under `uploads/updates/` and lets you select one to apply.

- **TLS Certificates** offers a **From SFTP exchange** source that lists the PEM files staged under `uploads/certificates/` and lets you pick a certificate/key pair to install.

A weekly cleanup timer removes files older than seven days from the exchange directory, so an upload that has been sitting around for a while will eventually be reaped. If something needs to stick around longer than that, consume it from the relevant TUI screen first.

Related screens

- SSH Access — manages operator-level shell access. Separate principal, separate key list.
- TUI Reference — the full menu, including the SFTP-source options on Update System and TLS Certificates.
- Installation — first-boot setup, where the appliance is provisioned but no SFTP keys are configured yet.

9.6 SSH Access

This page explains how shell access to the appliance is granted. SSH access is a vendor-support and file-transfer concern — not part of day-to-day operation. The terminal user interface (TUI) handles routine work directly on the console, and SSH exists only for the cases the TUI cannot cover.

Why SSH access exists

The Pulse appliance is designed so that 99% of operations are driven through the TUI. Operators install updates, manage TLS certificates, configure networking, and onboard SFTP keys without ever touching a shell. That keeps the system narrow and auditable.

There are two situations where shell access is still needed:

- **Vendor support emergencies.** When something goes wrong below the TUI — a kernel issue, a wedged service, a bug in the appliance image itself — a Pulse support engineer needs a shell on the appliance to investigate. SSH is the channel for that. The vendor signs a short-lived certificate for the support engineer's key; the operator does not have to provision anything in advance.
- **The `pulse-transfer` SFTP principal.** The keys the operator manages on the SFTP Access screen authorise an external party to drop files into the appliance's exchange directory. That party connects as `pulse-transfer` over the same SSH service, so SFTP delivery rides on top of the SSH stack described here.

In both cases the appliance trusts certificates signed by the appliance's SSH certificate authority (CA). There is no list of individual user keys to maintain on the appliance — trust is delegated to whoever holds the CA private key.

The two principals

Every signed certificate carries one or more *principals*. A principal is a label the SSH server uses to decide what kind of access the certificate grants. The appliance recognises exactly two:

`pulse` — operator role (SSH + TUI)

Logging in as `pulse` lands the user in the operator TUI. This is the same view the operator sees on the local console. From there they can manage networking, certificates, updates, and so on. The `pulse` user is **unprivileged**: they do not have a root shell, cannot edit system files directly, and cannot bypass the TUI to mutate appliance state. Anything they need to change goes through the menus, which call privileged wrappers under the hood.

Use `-n pulse` when signing a key for a support engineer who needs to operate the appliance the same way the on-site operator would.

`pulse-transfer` — SFTP only

The `pulse-transfer` principal authorises a chrooted SFTP session and nothing else: no interactive shell, no TCP forwarding, no X11, no tunnels. The holder of a `pulse-transfer` certificate can put files into `uploads/` and get files from `downloads/`, and that is the entire surface available to them. This is the principal used by the keys the operator adds on the SFTP Access screen.

Use `-n pulse-transfer` when signing a key whose only job is to move files in and out of the exchange directory.

A single certificate can carry both principals at once when the same person legitimately needs both roles — for example, a support engineer who will also upload an update bundle during the same session.

How a vendor signs a customer key

The signing happens **off the appliance**, on a workstation that holds the SSH CA private key. The appliance never sees the CA private key; it only carries the corresponding public key as a trust anchor.

The end-to-end flow:

1. The operator (or whoever needs access) generates an SSH keypair on their workstation if they do not already have one. The public half lives at `~/.ssh/id_ed25519.pub`.
2. The operator sends that `.pub` file to the vendor through whatever channel is convenient — email, ticket attachment, secure messaging. The private half never leaves the operator's machine.
3. The vendor signs the public key with the CA private key using `ssh-keygen -s`, choosing the principal that matches the access being granted.

4. The vendor returns the signed certificate file to the operator. By convention it is named `id_ed25519-cert.pub` (the same basename as the key, with `-cert` appended).
5. The operator drops the signed certificate next to their key — typically `~/.ssh/id_ed25519-cert.pub`. The SSH client picks it up automatically the next time it connects.

The signing commands the vendor runs are:

Operator role (`pulse` principal)

```
ssh-keygen -s /path/to/ca_private_key -I "description" -n pulse -V +365d ~/.ssh/id_ed25519.pub
```

SFTP transfer role (`pulse-transfer` principal)

```
ssh-keygen -s /path/to/ca_private_key -I "description" -n pulse-transfer -V +365d ~/.ssh/id_ed25519.
```

Both principals in one certificate

```
ssh-keygen -s /path/to/ca_private_key -I "description" -n pulse,pulse-transfer -V +365d ~/.ssh/id_ed2
```

In every case the signed certificate is written to `~/.ssh/id_ed25519-cert.pub` and picked up automatically by the SSH client.

The `-I` flag sets a free-text identity recorded inside the certificate — useful for audit trails. A short description like `support-eng-jane-2026-05` or the ticket number that prompted the signing is the right level of detail.

Validity windows

The `-V +365d` flag in the examples above caps the certificate's validity at one year from signing. After that, the certificate expires and the SSH server refuses to accept it — even though the underlying key is unchanged.

This expiry is **deliberate**:

- It limits the blast radius of a key leak. A stolen certificate stops working on its own; nobody has to find and revoke it.
- It forces a periodic re-signing ritual, which is also the natural moment to ask "does this person still need access?".

- It matches the operational rhythm of a managed appliance: a year is long enough that re-signing is not a constant chore, short enough that long-departed engineers do not retain access indefinitely.

Shorter windows are appropriate for one-off support sessions — `-V +1d` or `-V +1w` for a single intervention, after which the certificate is dead. Use shorter validity when the access is for a specific task; reserve the one-year window for keys that will be in use repeatedly.

The certificate's expiry is independent of the SSH key itself. The key stays valid; only the signature on it expires.

What the operator does not get

A signed `pulse` certificate gives the user a shell on the appliance, but that shell is the same constrained environment the local operator works in:

- The `pulse` user is **unprivileged**. They do not have `sudo` for arbitrary commands. They cannot edit system configuration files directly, cannot restart system services from the shell, and cannot read or write the appliance's protected secrets.
- Configuration changes go through the TUI. The TUI itself calls a small set of audited privileged wrappers to mutate appliance state. There is no shortcut around it.
- A `pulse-transfer` certificate is even more restricted: no shell at all, just SFTP into the exchange directory.

In other words: SSH access is a way to *reach* the operator role from off-site, not a way to *escalate* beyond it. Anyone who needs to change appliance state still does so through the TUI — they just see it over SSH instead of on the local console.

Related screens

- SFTP Access — manage the customer-facing SFTP keys that authenticate as `pulse-transfer`.
- TUI Reference — the operator menu the `pulse` user lands in after SSH login.
- Installation — first-boot setup, including the SSH CA trust anchor that makes signed certificates work.

9.7 TUI Reference

The Pulse appliance is operated through a terminal user interface (TUI) that opens automatically on the console (tty1) and over SSH. The main menu has twelve entries. Use the arrow keys (or `j` / `k`) to move between them, `enter` to open one, and `q` to quit.

System Status

Shows the appliance's software version, uptime, IP address, the health of each core service, disk usage, and when the last backup ran. Open this first when a user reports the application is slow, unreachable, or behaving strangely — it's the quickest way to confirm whether the box itself is healthy.

Network Configuration

Configures the appliance's IPv4 address, gateway, and DNS servers, and switches between DHCP and static-IP modes. Open this during initial setup, when moving the appliance to a different network, or when the LAN team changes the assigned IP.

NTP Configuration

Sets the time-synchronization server the appliance uses. Open this if the clock has drifted, if the site uses an internal NTP server instead of the default upstream pool, or after relocating the appliance to a network without internet access.

Update System

Applies a signed system update from a USB stick, the SFTP exchange, or a CD/DVD-ROM. Open this when a new Pulse release is announced, when a security update is required, or when on-site support has staged an update bundle for you to apply.

SFTP Access

Manages how customers authenticate for SFTP file exchange with the appliance — both the SSH public keys and the appliance's auto-generated password. Open this when onboarding a new customer integration, rotating a key or the password, or removing access when a relationship ends.

Backup / Restore

Creates a full database backup or restores the appliance from a previous backup file. Open this before risky operations (updates, configuration changes), on a regular cadence as part of normal operations, or to recover from data loss or a failed update.

View Logs

Browses recent system and application logs from journald and the Pulse server. Open this when diagnosing a service that won't start, investigating an alert that didn't fire, or gathering evidence to send to support.

Restart Services

Restarts the Pulse server, PostgreSQL, or nginx individually. Open this if a service has gotten stuck — for example, the web UI returns 502 errors but the appliance is otherwise healthy — and you want to bounce a single service rather than rebooting the whole box.

TLS Certificates

Manages the TLS certificate the web interface presents to browsers. Open this to switch between self-signed and customer-supplied certificates, generate a certificate signing request (CSR) for the customer CA to sign, or install a renewed certificate before the current one expires.

Rollback

Switches the appliance back to a previous system generation. Open this immediately after an update if the new version misbehaves — rollback is the safety net that lets the operator return to a known-good state without involving support.

Expert Shell

Opens an interactive system shell on the appliance, gated by a password. Open this only when support explicitly asks for it: it bypasses the TUI's guardrails and any change made here is invisible to the rest of the menu.

Reboot System

Reboots the appliance. Open this after an update that requires a kernel restart, when instructed by support, or as a last resort if the appliance is in a clearly broken state and the operator wants to start fresh.

9.8 System Updates

Pulse ships as signed update bundles produced by the vendor. The **Update System** screen in the appliance TUI applies one to the running appliance. The wizard is the same whichever way the bundle reaches the appliance — only the first step, the source, differs.

This page is the full walkthrough. For database backups and rolling back to a previous version, see Operations.

What an update bundle is

Every bundle is signed by Pulse. The appliance verifies the signature chain before it shows you anything, and refuses to proceed if the bundle is unsigned, tampered with, or signed by a key that does not match the public key baked into the running system. There is no way to apply an unsigned bundle from the TUI — that is the whole point of the signing chain.

A valid bundle carries a manifest with the version string, build timestamp, a short changelog, and a flag for whether the new version requires a reboot. You see all of this on the **Update Available** screen before you commit.

Choosing a delivery path

There are three ways to get a bundle onto the appliance. Pick by how the appliance is deployed:

Deployment	Recommended path	Why
Virtualized (VMware ESXi/vSphere, Proxmox, Hyper-V, …)	**CD/DVD-ROM**	Attach the update ISO to the VM's virtual optical drive — the same gesture you used to install. No USB passthrough to set up.
Physical / bare-metal appliance	**USB drive**	No hypervisor, so hand-carried removable media is the simplest path.

Deployment	Recommended path	Why
Remote / unattended, or staged by support	**SFTP upload**	Support delivers the bundle over the network; nobody has to walk to the rack.

All three feed into the same wizard and the same safety net described below — the table is about convenience, not capability.

How an update is applied

Once you confirm, the appliance runs a fixed sequence. It is worth knowing what happens, because most of it is automatic safety:

1. **Verify** — re-checks the signature and checksum.
2. **Back up the database** — takes a fresh PostgreSQL backup *before* touching anything, so the pre-update state is always recoverable.
3. **Import and switch** — unpacks the new system and switches the boot profile to it.
4. **Health check** — waits up to 60 seconds for Pulse to answer on its health endpoint.
5. **Auto-rollback on failure** — if the switch or the health check fails, the appliance rolls itself back to the previous version automatically and reports the failure. You are left on a working system, not a broken one.
6. **Clean up** — keeps the five most recent system generations and prunes older ones.

If the manifest says a reboot is required (typically a kernel change), the final screen tells you so; reboot from the main menu when convenient.

> **TIP**
>
> Because the appliance backs up the database and self-rolls-back on a failed health check, a bad update is largely self-correcting. The manual recovery tools on the Operations page are for problems that surface *after* the update is already running.

Before you update

A short checklist keeps an update boring:

- **Confirm the appliance is healthy now** — the web interface loads and the system is behaving. Updating a sick system makes diagnosis harder.

- **Make sure you can reach the console** — a local monitor and keyboard, or SSH as the `pulse` user. You will want to watch progress and reboot if asked.
- **Plan for a possible reboot** — some updates need one. Pick a window where a short restart is acceptable.
- **Have the bundle ready in the right place** for your chosen path (see each path below).
- *(Optional)* **Take a manual backup** from the Backup / Restore screen. The update takes one automatically, but an extra known-good snapshot is cheap insurance before a big jump.

Delivery paths in detail

Virtualized appliance — CD/DVD-ROM

This is the recommended path for any appliance running as a VM.

1. Copy the update ISO (`pulse-update-<version>.iso`) to your hypervisor's datastore.
2. Edit the VM's settings and point its **CD/DVD drive** at that ISO. In VMware vSphere/ESXi, set the drive to **Datastore ISO File**, browse to the ISO, and — this is the step people miss — tick **Connected** (and **Connect At Power On**). Other hypervisors have the same two ideas: choose the ISO, and make sure the drive is actually connected.
3. On the appliance, open **Update System → CD/DVD-ROM**. The TUI mounts the virtual drive, finds the bundle, and verifies it.

> **WARNING**
>
> If the TUI reports no disc, the drive is almost always attached but not **Connected**. Detection keys off the disc being mounted, not merely present in the VM's configuration. Tick **Connected** and retry.

Do not disconnect the disc while the update is running — the **Updating…** screen warns about this in yellow. The drive is unmounted automatically when the update finishes.

Physical appliance — USB drive

1. Format a USB stick as **FAT32** or **ext4**.
2. Copy the bundle files to the **root of the drive** (a single top-level folder such as `updates/` also works). Do not bury them deeper.
3. Plug it into the appliance and open **Update System → USB Drive**. The TUI scans for removable drives:

- One drive found → it mounts and scans automatically.
- More than one → the **Select USB Device** step lets you pick the right one.
- None → it reports `no USB drive found (format must be FAT32 or ext4)` and waits.

Do not unplug the stick while the update is running — the bundle is read from it as it is applied, and the **Updating…** screen warns in yellow. It is unmounted automatically on completion.

Remote / support-staged — SFTP upload

1. The party delivering the bundle must already be an authorised SFTP user — see SFTP Access to authorise them, either by their SSH key or with the appliance's SFTP password.
2. They upload the bundle into the exchange directory under `uploads/updates/`.
3. On the appliance, open **Update System → SFTP Upload**. The TUI scans the exchange directory and verifies what it finds.

A successfully applied SFTP bundle is removed from the exchange directory afterwards, so it cannot be applied twice.

The wizard, step by step

Whichever source you pick, the rest is identical:

1. **Source** — pick USB Drive, SFTP Upload, or CD/DVD-ROM. The TUI scans, mounts (USB and CD/DVD-ROM), and verifies the bundle.
2. **Manifest review** — the **Update Available** screen shows the version, build timestamp, changelog, and whether a reboot is required. Read it. Press `Esc` to back out cleanly without applying anything.
3. **Confirm** — press `Enter` on the manifest screen to start. There is no second yes/no prompt; the manifest screen *is* the confirmation.
4. **Progress** — the **Updating…** screen streams progress as the bundle is verified, backed up, imported, and activated. If removable media is still mounted, it reminds you in yellow not to remove it.
5. **Done** — the final screen shows `Update Successful!` (green) or `Update Failed` (red) with the underlying error. Press `Esc` to return to the main menu.

After the update

- If the manifest flagged a reboot, reboot from the main menu's **Reboot System** entry.
- Confirm the web interface comes back and the version is the one you expected.

- Once you are satisfied, take a fresh backup from Backup / Restore so your next known-good snapshot is post-update.

Troubleshooting

What you see	Likely cause	What to do
`no USB drive found (format must be FAT32 or ext4)`	Stick isn't FAT32/ext4, or wasn't detected	Reformat as FAT32 or ext4, reseat the stick, retry
USB mounts but no bundle is found	Bundle isn't at the drive root, or files are incomplete	Put `manifest.json` + `system.tar` (and the `.sha256` / `.sig` files) at the **root or in a single top-level folder** (e.g. `updates/`) — do not nest deeper
CD/DVD-ROM: no disc detected	ISO not attached, or attached but not **Connected**	Attach the ISO, tick **Connected** / **Connect At Power On**, retry
SFTP: no bundle found	Not uploaded to `uploads/updates/`, or sender not authorised	Confirm the upload path; authorise the sender (SSH key or password) via SFTP Access
Verification / signature error	Bundle is unsigned, corrupted, or signed with the wrong key	Get a correctly signed bundle from Pulse; re-applying the same file won't help
Update Failed during apply	Switch or health check failed — the appliance already auto-rolled-back	You're back on the previous version. Capture the error and contact support with it
Web app unreachable after a reboot	New version is unhealthy	Roll back to the previous generation, then restore a backup if needed

When an update goes wrong

A failed update auto-rolls-back during apply (see the troubleshooting table above). For problems that surface after the update is running, use Rollback to return to the previous system generation. Rollback is system-level only — it never touches the database or

other data under `/var`. See Operations for backups, restore, and rollback in full.

9.9 Operations

This page covers the operator procedures for protecting and recovering appliance data: taking and restoring database backups, and rolling back to the previous system generation if an update misbehaves. Applying a system update has its own page — see System Updates. Each section is anchored to one TUI screen, so you can match the on-screen wizard to the steps described here.

Updating the system

Applying a signed update bundle has its own page — see System Updates for the full walkthrough: which delivery path to use for virtualized versus physical appliances, the built-in backup-and-rollback safety net, and troubleshooting.

Backup and restore

The **Backup / Restore** screen manages database backups. The appliance takes automatic backups on a schedule; you can also create an ad-hoc one before risky operations, and you can restore from any backup the screen lists.

What you see

The top-level menu has two entries:

- **Create Backup** — runs a backup immediately. The screen shows a green status line with the output of the backup command on success, or a red error on failure.
- **List / Restore Backups** — opens the picker described below.

Picking **List / Restore Backups** shows every backup file the appliance has on disk, one per line, with the file name, size (KB, MB, or GB), and last-modified timestamp. The most recently modified backup is the one you usually want when recovering from a problem; the older entries are the rolling automatic backups.

Creating an ad-hoc backup

Before doing anything risky — an update, a large configuration change, a customer-supplied import — pick **Create Backup** and wait for the green confirmation. The result is a fresh backup file alongside the automatic ones, and it gives you a known-good rollback point for the database that does not depend on the next scheduled backup running.

Restoring a backup

Restoring overwrites the live database with the contents of the chosen backup file. The screen takes that seriously:

1. From **List / Restore Backups**, move the cursor to the backup you want to restore and press `Enter`.
2. A confirmation screen appears with a yellow warning — `WARNING: This will replace the current database!` — and the name of the backup file. Press `y` to confirm or `n` / `Esc` to back out.
3. On confirmation, the restore runs and the screen shows `Database restored successfully` (green) or the underlying error (red).

While the restore is running the database is being replaced wholesale: any rows written since the chosen backup are lost, and any user logged into the web interface will see their session blow up the moment the database goes away. Restoring is not a partial or merge operation — it is a full replacement, which is exactly why the warning is loud.

Backup vs rollback

Backups cover the *database*. The system itself — the NixOS generation, configuration, installed software — is covered by Rollback. The two are complementary: a backup will not bring back an old kernel, and a rollback will not bring back deleted records. When in doubt about which one you need, see the next section.

Rollback

The **Rollback** screen switches the appliance to a previous system generation. Use it when an update misbehaves and you need the old version back fast.

What rollback does

The screen lists every NixOS generation the appliance retains, newest first, with its number, build date, and a green `(current)` marker on the one that is running. Move the cursor with the arrow keys (or `j` / `k`), press `r` to refresh the list, and press `Enter` on a non-current generation to start the switch.

A confirmation prompt appears asking `Switch to generation N (date)?`. Press `y` to commit or `n` / `Esc` to back out. On confirmation the appliance updates its boot configuration and reports `Switched to selected generation`. **A reboot is required for the change to take effect** — the running services are still on the old (now-newer) generation until the box restarts. Reboot from the main menu's **Reboot System** entry once the rollback is confirmed.

What rollback preserves — and what it doesn't

Rollback is a system-level operation, not a data-level one. Concretely:

- **Replaced**: the root partition. The Nix store, kernel, system configuration, and installed software all revert to what the previous generation looked like.
- **Preserved**: `/var` and everything below it. That includes the Pulse PostgreSQL database, your database backups, log files, the SFTP exchange directory, and any configuration written through the TUI (TLS certs, SFTP keys, network settings). None of that is touched by switching generations.

In practice, rolling back undoes the *software* part of the last update but keeps every byte of customer data the appliance has accumulated since then.

When to roll back vs. restore a backup

The two recovery tools cover different failure modes; picking the right one matters.

- **Roll back** when the new software is the problem: a service that won't start, a regression in behaviour, a kernel issue introduced by the update, a TUI screen that crashes. The data is fine; the code is not. Rollback restores the previous generation without losing any rows in the database.
- **Restore a backup** when the data is the problem: corruption, an accidental mass delete, a bad import that polluted the schema. The code is fine; the data is not. Backup restore returns the database to a known-good snapshot.

If both are needed — an update went wrong *and* corrupted state on the way out — roll back first to get the appliance onto a working software version, then restore the most recent pre-update backup.

Generations that no longer appear

NixOS only retains a bounded number of generations on disk. Older generations are garbage-collected on the appliance's normal schedule, so the list does not grow forever. If the generation you want to roll back to is no longer listed, you are past the point where rollback can help and the appliance needs to be re-imaged or repaired via support — the rollback screen is a short-term safety net, not a long-term recovery archive.

After a successful rollback

Once the appliance has rebooted onto the previous generation and you have confirmed the system is healthy again, take a fresh database backup from the Backup and restore screen. The post-rollback state is now your known-good baseline; capturing it explicitly means the next risky operation has a clean snapshot to fall back on instead of the older automatic backups that predate whatever you just recovered from.

If the update that triggered the rollback was delivered over SFTP, the bundle has already been cleaned up by the post-update flow on the *original* (failed) attempt. If it was delivered over USB, the stick is no longer mounted — pull it out, work out with support what was wrong with that bundle, and only retry once a corrected build is available. If it was delivered via CD/DVD-ROM, the drive is automatically unmounted and you can safely disconnect the ISO from the VM. There is no value in re-applying the same bundle that failed last time.