Permissions and Personas

OrgGuardian customer guide — who sees what, and by which right.

> **The version installed in your org can be read in Setup → Installed Packages**, and what each permission set grants in **Setup → Permission Sets**: this page is the guided reading of those sets; it shows you how to put them to use. Object, field and custom metadata type API names are given here without the `orgguardian__` namespace prefix that they carry in your org.
>
> **No permission set of the package declares a system permission**: a permission set installed from a package does not carry user permissions. The scan account, which runs the scans, needs four read rights, and you grant them: § 2, "The scan account: four read rights that you grant".

1. The five shipped permission sets

The package ships **five** permission sets, no more, no less. In **Setup → Permission Sets**, they appear under the labels of the table below.

In a subscriber org, each one carries the namespace prefix: `orgguardian__OrgGuardian_Viewer`, `orgguardian__OrgGuardian_Analyst`, etc. It is in this form that the persona groups reference them.

Overview

API name	Label	Objects opened	Apex classes	Visible tabs	App	Custom permission
`OrgGuardian_Viewer`	OrgGuardian: read-only access	27 (read-only)	49	35 / 43	`OrgGuardian`	—
`OrgGuardian_Analyst`	OrgGuardian: finding handling	27	53	35 / 43	`OrgGuardian`	—
`OrgGuardian_Auditor`	OrgGuardian: audit and compliance	27	50	35 / 43	`OrgGuardian`	—
`OrgGuardian_Admin`	OrgGuardian: technical administration	32	131	43 / 43	`OrgGuardian`	`OrgGuardian_Manage_Features`, `OrgGuardian_Run_Incident_Drill`
`OrgGuardian_Financials`	OrgGuardian: financial amounts	0	0	0	—	`OS_View_Financials`

Each set's page in **Setup → Permission Sets** shows these grants: **Object Settings**, **Apex Class Access**, **Visualforce Page Access**, **Custom Permissions**, **System Permissions** (none is enabled in any of the five sets) and, for `OrgGuardian_Admin` alone, **Custom Metadata Types** and **External Credential Principal Access** (§2).

Object lines. Viewer, Analyst and Auditor each carry **28** object lines, **27** of them opened: the line of `Login_Baseline__b`, the per-user login habits, grants no access, not even Read, and only `OrgGuardian_Admin` reads that object. `OrgGuardian_Admin` carries **32**: the 27 objects of the other sets, `Login_Baseline__b`, the telemetry event `Callout_Log_Event__e` and three platform configuration objects read by the scanners (§2). The fleet inbox, `Fleet_Finding__c`, is in the four sets (read for Viewer and Auditor, read and edit for Analyst, create, read, edit and delete for Admin), as is `Config_Baseline_Item__c`, the setting-level detail of the signed baselines (read for the three non-admin sets, create and read for Admin), and `OrgGuardian_PurgeCursor__c`, the purge's resume anchor, with its `Cursor__c` field, which the chained-ledger check reads back after a retention purge (read for the three non-admin sets, create, read, edit and delete for Admin).

Apex classes. 49 / 53 / 50 / 131 (Viewer / Analyst / Auditor / Admin). Analyst adds to the classes of Viewer the finding actions, ticket creation, the CSV export and the Agentforce posture actions; Auditor adds the CSV export alone. The four sets open, among others, the **My alerts** screen, the fleet inbox, the criticality scope line of a finding sheet and the user access review, and read the compliance status of an SLA report (`Org_Sla_Report__c.Compliance_Status__c`); Admin alone opens the editors, among them the fleet policy editor, the criticality scope editor and the Tooling API access assistant.

Visualforce page. Viewer, Analyst, Auditor and Admin open the `OrgGuardianBoardPack` page, which renders the board pack as a PDF.

Field permissions. Viewer, Analyst, Auditor and Admin each carry **239** field permissions, almost identical from one set to another. For reading, `OrgGuardian_Viewer` is denied twelve fields: a finding's target, message and message arguments, its business unit, the name of the criticality scope that moved its severity and the reference of the Slack message posted for it (`Chat_Message_Ref__c`); the incident post-mortem; the three fields of the login baseline; the hashed reference of a fleet inbox line, `Remote_Ref__c`; and `Callout_Log__c.Caller_Id__c`, the Id of the user whose code made an instrumented callout — an observed person —, which the three other sets read. `OrgGuardian_Analyst` is denied five: the three login-baseline fields, `Remote_Ref__c` and `Chat_Message_Ref__c`. `OrgGuardian_Auditor` is denied four: the three login-baseline fields and `Remote_Ref__c`. `OrgGuardian_Admin` reads all of them. What really separates the sets is **object-level CRUD**, **Apex class access** and **tabs**.

If you build your own permission sets. A set that opens the **Approved baseline** screen must also grant read access to `Config_Baseline__c.Seal_Version__c`: without it, the screen says “The approved baseline could not be read” instead of comparing. To read each deviation setting by setting, it also needs read access to the `Config_Baseline_Item__c` object and to its `State_Facets__c` field: without it, the deviation is read component by component, and the screen says so. Approving a baseline also requires create access to `Seal_Version__c`; without create access to `Config_Baseline_Item__c` and its two fields, the baseline is approved without its detail, and the screen says so. The screen also reads `Config_Snapshot__b.State_Facets__c`, which the time machine and the parity comparison already read: without that read access, the screen compares nothing and says “Reading the current state was refused”. The user who schedules the board report needs, besides create access to `Posture_Report__c`

and its four content fields, create access to its two fingerprint fields, `Archive_Hash__c` and `Prev_Hash__c`: without it, the archive is still stored, but without a fingerprint, and it reopens as **no fingerprint**. Checking an archive when it is downloaded needs read access to these two fields: without it, the verdict is **not checked**.

---

2. What each set opens, in detail

`OrgGuardian_Viewer` — read-only access

Shipped description: "Read-only access to OrgGuardian screens: dashboards, findings, compliance and limits. No writes, no scan triggers, no settings editors. Withheld from this set: finding target, message and its arguments, login baselines, incident post-mortem."

Object rights — Read only, on 27 of the product's objects, without exception: `Finding__c`, `Insight__c`, `Config_Dependency__c`, `Incident__c`, `Org_Sla_Report__c`, `Itsm_Ticket__c`, `Org_Metric_Current__c`, `System_Log__c`, `Scan_Run__c`, `Linked_Org__c`, `Fleet_Finding__c` (the fleet inbox), `Callout_Log__c`, `Attestation__c`, `Config_Baseline__c`, `Config_Baseline_Item__c` (the setting-level detail of the signed baselines), `Finding_Activity__c`, `Posture_Report__c`, `OrgGuardian_PurgeCursor__c` (with its `Cursor__c` field, the anchor that the chained-ledger check reads back after a retention purge), `Endpoint_Status__c`, `Discovered_Endpoint__c`, the platform event `Org_Health_Alert__e`, and six of the seven trend Big Objects: `Config_Snapshot__b`, `Em_Metric_Snapshot__b`, `Compliance_Snapshot__b`, `Prediction_Outcome__b`, `Endpoint_Health_Snapshot__b`, `Org_Limit_Snapshot__b`. The seventh, `Login_Baseline__b`, is not read by this set (nor by Analyst and Auditor: only `OrgGuardian_Admin` reads it): 28 object lines, 27 of them read.

Tabs: 35 of the 43. The **eight** administration tabs are hidden from this set: **Administration**, **Settings**, **Subscription**, **Alert routing**, **Connections**, **Alert channels**, **Enablement** and **Incident drill** — each drill launch writes an audit record in the org, and its live mode really notifies the on-call rota. Lightning responds to them with "Page doesn't exist": this is the intended behaviour, not an anomaly.

No user permissions: this set carries neither `ViewSetup`, nor `ViewAllUsers`, nor `ViewRoles`. That is what makes it assignable to a *Salesforce Platform* licence. In Setup, its **System Permissions** section has none enabled.

What read-only access triggers outside the org. Two reads of the Fleet and Parity screens call a linked org, on demand, in the reader's session — under this set too: the findings drawer of a linked org (feature **Linked org findings**) and the parity comparison with a linked org or with the reference org (feature **Fleet parity**). Both are paid-plan features, off at installation. Each call goes through the Named Credential of the registry row: a reader needs access to the principal of its External Credential, and the linked org sees that principal's identity. Nothing is written in the linked org, but each click uses one request of its API quota (two for the comparison with the reference org). The parity comparison shows the reader the configuration of the linked org, including which profiles and permission sets grant sensitive rights (Modify All Data, View All Data, Author Apex, Manage Users…) and the three policy settings of each connected app — never a person. Designating the reference org and running the sweep that compares the whole fleet require edit rights on the registry and its parity fields: `OrgGuardian_Admin` only.

What read-only access reads from your linked orgs without calling them. This set also reads the fleet inbox (`Fleet_Finding__c`, same **Linked org findings** feature): the open findings of your linked orgs, kept in the managing org for triage — severity, category, source, occurrences, dates and triage status; never their target field, their message or the reference of the linked org's record; the source, for its part, names the finding's target for some families (a host, a metric, an audited setting), never a user — a source that names one is not kept. These lines were read under the identity of whoever synced the inbox — an administrator, or, every 4 hours, the user who scheduled the fleet sync — and this set reads them whether or not it has access to the Named Credential's principal. The history of the triage status says who triaged a line, and when: an operator identity, visible by design.

What read-only access shows of the fleet conformance. Behind the **Multi-org aggregation** and **Fleet conformance** features (paid plan, off at installation), this set reads on the Fleet screen the verdict of each linked org — conforms, does not conform or not evaluated —, the policies exceeded or not evaluated (thresholds, counts, ages) and the "N of M orgs conform" count, and can export them as CSV: a local download, nothing sent. It reads neither the label nor the rationale of a policy and changes none: the fleet policy editor is reserved to `OrgGuardian_Admin`.

What read-only access exports of the configuration. This set can download the CSV export of the configuration time machine (**Configuration drift** tab: the state at a date, or the difference between two dates). The **Config state at a past date** feature (paid plan) is off by default. The file contains configuration only: component types and API names, setting = value pairs, names typed in your org included (threshold keys, features created in your org), which the screen already shows; the per-user effective-permission registry never enters it. It is built on request in your org and saved by your browser: a local download, nothing sent.

What read-only access shows about your users — read before assigning

This set is read-only, but **it is not entirely anonymous**. Several findings carry a user's identity:

- **on every plan, `FREE` included**: failed logins **per user** (who, how many) and, through the free security core, human accounts without MFA, the holders of dormant OAuth authorizations and the dormant or deactivated owners of a live token;
- **with the full Security pack (paid plan)**: privileged accounts and their hygiene (inactive account, overly broad rights, password that never expires), access by deactivated users, login anomalies and the dispersion of rights;
- **with the board report (paid plan, and during the 30-day trial of the full plan)**: the top 5 of the board pack built on demand. Its detail column can name the user that a finding cites, but only for a reader of the finding's target and message: for this set, it prints a neutral text. The weekly archive has no detail column: it carries neither target, nor message, nor name.

Target, message and message arguments withheld. This set does not read the **target**, the **message** or the **message arguments** of a finding (`Finding__c.Target__c`, `Finding__c.Message__c` and `Finding__c.Message_Args__c`: read access withheld from this set only; Analyst, Auditor and Admin keep it). Where a screen displays the name or the message, this set sees "Detail restricted for your access level" instead, and the board pack built on demand prints a neutral text in place of the detail. It still sees the number, the severity and the category of findings. The same withholding covers the **post-mortem** of a resolved incident (`Incident__c.Postmortem__c`), which copies the message or the target of its findings: no screen of this set shows it.

What remains readable by this set, and can identify a person:

- a finding's deduplication key and measured value (`Dedupe_Key__c`, `Metric_Value__c`), an incident's key (`Incident_Key__c`) and its **root cause** (`Root_Cause__c`), which copies that key, and four fields of the toxic-combination insights (`Insight__c.Subject_Key__c`, `Dedupe_Key__c`, `Target_Setting__c`, and `Recommended_Action__c`, whose text cites the account concerned): they can carry a user's opaque identifier and, for failed logins, their number;
- the lines of the **alert dispatch log** (`System_Log__c.Message__c`), which copy the key of the finding alerted on;
- the line of the **configuration log** that each editor writes (`System_Log__c.Message__c`, "Configuration changed (job …): `Type__mdt.DeveloperName`"), and the same key on an incident sheet (a correlated OrgGuardian setting change): a record appears there under its API name — the name you gave it if it was created in Setup, and, if an editor created it, a name derived from what you typed: the name entered in the **Sensitive identities** pane, the label entered in the **Routing rules**, **Risk accepted**, **Maintenance windows** and **Endpoint tagging rules** panes, the metric key entered in the **Custom rules** and **Per-metric thresholds** panes. Only a criticality scope created by the editor carries an opaque API name (`Scope_` followed by sixteen hexadecimal characters);
- the list of **routing rules** (name, label, channel, route) and the channel of a dispatch in the log: a rule that names a business unit or a criticality tier, with a channel of its own, ties the pseudonym of a finding to that unit or tier;
- the user keys of the **configuration graph** (`Config_Dependency__c`, paid plan, off by default) and of the **effective-rights register** (`Config_Snapshot__b`).

By design, operator identities remain visible: who took charge of a finding, accepted a risk, signed a control, and triaged a fleet inbox line.

This set does not read the **business unit** of a finding either (`Finding__c.Business_Unit__c`, the Division of the user concerned: the business unit of a single person identifies them), nor the **name of the criticality scope** that moved its severity (`Finding__c.Criticality_Scope__c`); it reads the severity the check reported (`Finding__c.Base_Severity__c`), and the finding sheet tells it that a scope lowered or raised the severity, without naming the scope.

By assigning this set, you therefore still give its holder **access to the identity of the users concerned**, through these fields. Reserve it for people with a need to know the security posture of your users.

Exports. Once the **Config & posture export** feature is on (paid plan, off by default), this set can also download the posture as JSON or CSV. The export leaves out each finding's target and message, but keeps its deduplication key (`Dedupe_Key__c`), which, for a finding about a single user, identifies that user.

`OrgGuardian_Analyst` — finding handling

Viewer, plus the finding lifecycle. The only write rights added:

Object	Rights
`Finding__c`	Read + **Edit**
`Finding_Activity__c`	Read + **Create** (neither Edit nor Delete)
`Itsm_Ticket__c`	Read + **Create** + **Edit**
`Posture_Report__c`	**Read** only, as in Viewer (no write right added): no action of the product modifies a board report archive, each archive carries a fingerprint checked when it is reopened, and the scheduled board report, run under `OrgGuardian_Admin`, is its only writer; a row inserted ahead of time, dated a day to come, would take the place of that day's sealed archive
`Fleet_Finding__c`	Read + **Edit**; only the triage status of a fleet inbox line is editable

On screen, this set opens the **Finding actions** and **Bulk actions** menus of the findings screens — **Assign to me**, **Unassign**, **Snooze…**, **Acknowledge…**, **Accept risk…**, **Create ticket…**, described in the Operations Guide (§3) — and, like `OrgGuardian_Auditor`, the **Export CSV** button.

`OrgGuardian_Auditor` — audit and compliance

Viewer, plus **a single** write right: `Attestation__c` with **Read + Create**, without Edit or Delete. That is what the **Attest** and **Plan the remediation** actions of the **Compliance** screen require; both also need the **DORA / NIS2 pack** feature, on by default, and a paid plan. The auditor modifies **no** finding, runs **no** scan, opens **no** editor.

`OrgGuardian_Admin` — technical administration

Shipped description: "Technical bundle of the scan account: write access to OrgGuardian's records. Setup read access comes from a permission set you create. Toggling features also needs "Modify Metadata Through Metadata API Functions", muting "Customize Application"." The read access to configuration and users that the scans need does not come from this set: it comes from a permission set that you create (below). This is not a business-administrator persona: it is the set that **runs the collectors**. On its own, the set does not let a user enable or mute a feature: that save also needs a platform privilege that the set does not grant (see its custom permissions below).

- **Full CRUD** on the objects produced by the product: `Finding__c`, `Insight__c`, `Incident__c`, `Org_Metric_Current__c`, `System_Log__c`, `Itsm_Ticket__c`, `Org_Sla_Report__c`, `Config_Dependency__c`, `Posture_Report__c`, `Endpoint_Status__c`, `Discovered_Endpoint__c`, `OrgGuardian_PurgeCursor__c`, `Fleet_Finding__c` (the fleet inbox: the sync creates, updates and removes its lines), and three trace tables that the retention purge also empties, `Scan_Run__c`, `Callout_Log__c`, `Finding_Activity__c`: the purge requires Delete, and the platform does not grant Delete without Edit.

- **Create + Read only** (registers that a user neither modifies nor deletes): `Attestation__c`, `Config_Baseline__c` (the approved configuration baseline: promoting a new one replaces the previous one without ever modifying it), `Config_Baseline_Item__c` (its setting-level detail, written once at approval), and the seven Big Objects, which only the retention purge erases on expiry. The platform event `Org_Health_Alert__e` is likewise opened with Create + Read, as is `Callout_Log_Event__e`: the right to publish the telemetry of `OrgGuardianCallout.execute`, which the package publishes instead of writing `Callout_Log__c` in the caller's transaction. `Linked_Org__c`: Read + Create + Edit, without Delete.
- **Read-only** on three platform configuration objects: `NamedCredential`, `CspTrustedSite`, `ExternalDataSource` — these are sources scanned by the connectivity posture detectors.
- **No system permission.** A permission set installed from a package does not carry user permissions: in an org where the package is installed, this set, like the eight other sets and groups of the package, has none enabled. The scan account, which runs the scans, does need four read rights: see "The scan account: four read rights that you grant", at the end of this section.
- **Read access to three Custom Metadata Types**: `Attack_Path_Rule__mdt` (attack path rules), `Acceptance_Policy__mdt` (risk acceptance policies) and `Email_Alert_Template__mdt` (email alert templates) — the only accesses to a metadata type carried by any set in the package. These three types are read with the user's own rights, which requires the user to see the type: the attack path analysis reads its rules this way, the acceptance policy editor and the rule pack export read the full reason of a policy, and the alert channel editor reads the full text and recipients of an email template.
- **Access to the principal of the packaged External Credential `OS_SelfOrg`** (`OS_SelfOrg-SelfOrg`): the calls that the product makes to your own org's API go through it. These calls are off by default.
- **Two custom permissions**: `OrgGuardian_Manage_Features` ("OrgGuardian: manage features"), which OrgGuardian checks before an administration action such as a configuration save — on its own, with the class access this set grants, it already opens, among others, the configuration export, the Heroku viewer link, manual publishing and the connection test, an alert channel test (a real send), the incident drill's live mode and reading the alert preferences of other recipients — and `OrgGuardian_Run_Incident_Drill` ("OrgGuardian: incident drill"), which allows launching an incident drill in rehearsal mode and reading the scenario catalogue without granting admin authority; live mode, which really sends to the on-call rota, still requires that authority. The first does not replace the platform privilege that a Custom Metadata or protected-setting save itself requires, and this set grants no such privilege ("Who is allowed to save", in the Configuration Guide).
- **43 tabs out of 43** — the only set that sees the eight administration tabs listed under `OrgGuardian_Viewer`.

The scan account: four read rights that you grant

Scheduled analyses run as the user who armed them: the owner of the "OrgGuardian Hourly Scan" job (Setup → **Scheduled Jobs**, "Submitted By" column), that is, the first authorised user who opened the OrgGuardian home screen. "Run scan now" runs as the user who clicks. This scan account needs four system permissions, all **read-only**, that **no set in the package can grant**: a packaged permission set does not carry user permissions.

- **A system administrator already holds them** through the profile: nothing to do.

- **Otherwise**, create a permission set (no licence), tick the four system permissions of the table, and assign it to the scan account **before** the OrgGuardian home screen is opened for the first time.

Permission (Salesforce label)	API name	What is missing without it
View Setup and Configuration	`ViewSetup`	Salesforce refuses configuration reads (permission sets and their assignments, setup audit trail, scheduled jobs): the security controls that depend on them are not analysed, and the refusal is reported as a refused read.
View Roles and Role Hierarchy	`ViewRoles`	Salesforce does not grant View Setup and Configuration without it. It opens the role hierarchy: exposure through the hierarchy, recipients of an alert addressed by role.
View All Users	`ViewAllUsers`	If user sharing is restricted in your org, the population read is truncated **without any error**: accounts without MFA, dormant privileged accounts and guest users may be missing from the analysis.
View All Profiles	`ViewAllProfiles`	When profile filtering is on in your org, other accounts' profiles are unreadable **without any error**: privileged and integration accounts are no longer recognised, and the other profile-based reads come back incomplete. If your org does not offer this permission, tick the other three.

OrgGuardian reads these rights at the start of every scan, under the account that runs it: a missing permission is named by a read notice (Security category, High severity, key `OrgGuardian:ScanRights:<API name>`), the Security score is reported as partial, and the controls that depend on it are not evaluated, neither credited with a pass nor closed as fixed; a finding they still raise stays open and counts as a failure. The "Give the scan account read access to setup" step of the setup tour checks the same rights on the scan account (the Installation Guide § 5.1 bis). Grant them before the first scan. Other, optional rights enrich the analysis (logins of all users, OAuth tokens, MFA methods): § 4.

`OrgGuardian_Financials` — amounts in euros

Zero object rights, zero field rights, zero Apex classes, zero tabs. The set, labelled "OrgGuardian: financial amounts", carries a single grant: the custom permission `OS_View_Financials` ("View financial amounts"). Custom permissions **add up** across assigned sets: for a persona **not** to see euro amounts in the default "Permission required" mode (§5), it is sufficient — and necessary — that none of its sets carries this permission. That is why it lives in a dedicated set and **in no other**.

---

3. The four personas (permission set groups)

What the administrator assigns day to day are the **groups**, not the sets:

Persona	Group API name	Actual composition	Euro amounts
Executive leadership / executive committee	`OrgGuardian_Persona_Direction`	Viewer + Financials	yes
IT department	`OrgGuardian_Persona_DSI`	Viewer + Financials	yes
CISO	`OrgGuardian_Persona_RSSI`	Viewer + Auditor + Financials	yes
Salesforce admin	`OrgGuardian_Persona_AdminSF`	Viewer + Analyst + Admin	**no**

Three group API names use French terms: *Direction* stands for executive management, *DSI* for the IT department and *RSSI* for the CISO.

The persona ↔ permission set mapping is exact. In **Setup → Users → Permission Set Groups**, the four groups appear as "OrgGuardian: executive leadership persona", "OrgGuardian: IT department persona", "OrgGuardian: CISO persona" and "OrgGuardian: Salesforce admin persona"; each group's **Permission Sets in Group** page lists exactly the composition of the table above.

Executive leadership and IT department currently have the **same composition**. This is deliberate: two distinct business intents, free to diverge later without breaking assignments already made.

`OrgGuardian_Persona_AdminSF` does **not** carry `OrgGuardian_Financials`: a Salesforce administrator will therefore not see euro amounts in the "Permission required" mode, which is the default (see §5).

Assigning a persona

OrgGuardian **never** assigns anything itself. The administrator does it in Setup: **Setup → Users → Permission Set Groups** (`/lightning/setup/PermSetGroups/home`) → open the group → **Manage Assignments**.

After installation, check that each group's status is `Updated`. The platform recalculates groups asynchronously: a status that remains stuck is diagnosed **before** any assignment.

Departing from the shipped composition

Three levers, in this order of preference:

1. **Assign the sets individually** (e.g. an IT department persona without euro amounts = `OrgGuardian_Viewer` alone).
2. **Create a local group** in the subscriber org that combines the packaged sets differently.
3. **A local muting permission set** attached to the installed group, to switch off an inherited permission.

Never modify the contents of the packaged permission sets: compose locally; do not mutate what is managed.

Sends that follow your permission sets

Two features choose their recipients by rights, and those rights are read on **permission sets**, never on a persona:

- **The reports by recipient** (**Reports by recipient** feature, `Report_Routing`, paid plan, off by default; **Posture digest** pane, the Configuration Guide §3) are emailed to the **active internal** members of the permission set that an executive or CISO template names (`Report_Template__mdt.Recipient_Permission_Set__c`); external members (Experience Cloud, partners) are left out and counted. Each block of the report is sent only if **that permission set itself** grants read access to the block's sources (object permissions, field permissions on each field that the block shows or counts, under a public internal sharing model) — not the rights that a member holds from another permission set; a permission set that grants none receives nothing. The board pack (**Board report** feature on) goes only to a permission set that reads the sources of the executive screen and, while it carries euro amounts in the "Permission required" mode, grants `OS_View_Financials` (§5). `OrgGuardian_Viewer` grants read access to all of these sources, but not `OS_View_Financials`: no packaged permission set combines both, and a board pack that carries euro amounts goes only to a permission set of your own that grants both. **Personas are groups**: naming a permission set finds only the users to whom it is assigned **directly**, not those who hold it through a persona; assign the named permission set to the recipients. Its API name must be carried by that permission set only: a name carried by more than one is refused when saved, and shown on the card if it becomes ambiguous later. The recipients of one message see each other in its "To" header (at most 150 per message), as for every email alert of the product; a report goes to **1,500 addresses at most** (ten messages): beyond that, to the first 1,500 members in the order of their user Id, and the `ALERT_EMAIL` log says so. Each address counts against your org's daily email limit.
- **The remediation reminders** (**Remediation reminders** feature, `Remediation_Reminders`, paid plan, off by default, with the DORA / NIS2 pack) ring only for an **active internal** owner who **actually reads** the register: the `Attestation__c` object, its rows (public internal sharing model, `Read` as shipped) and the three fields that the bell shows (`Framework__c`, `Article__c` and `Due_Date__c`), through their permission sets or their profile — a permission set that requires session activation does not count. `OrgGuardian_Viewer`, `OrgGuardian_Analyst`, `OrgGuardian_Auditor` and `OrgGuardian_Admin` grant them. Under a private sharing model, no bell is sent, and the log says so. When personal alert preferences are on (`Alert_Self_Service`), the owner's own apply to the bell (threshold, where the reminder counts as Medium; quiet hours in their time zone).

The **Since the previous scan** card at the top of the **Dashboard** screen (**Scan narrative** feature, `Scan_Narrative`, on by default) is shown to every persona that opens that screen, Viewer included: it carries only scores, counts and categories, never a finding's target, a name or an Id.

---

4. What no permission set opens

The product has a **read-only / detect and advise** posture: it does **not** remediate or write to customer business data. In concrete terms, across the five sets combined:

- **No `ViewAllRecords`, no `ModifyAllRecords`.** You can check it in your org with a SOQL query (for example in the Developer Console query editor), which returns no rows:

  ```
  SELECT Parent.Name, SobjectType FROM ObjectPermissions
  WHERE Parent.NamespacePrefix = 'orgguardian'
    AND (PermissionsViewAllRecords = true OR PermissionsModifyAllRecords = true)
  ```

- **No "Modify All Data", "View All Data", "Author Apex" or "Manage Users" permission** — this query returns no rows either:

  ```
  SELECT Name FROM PermissionSet
  WHERE NamespacePrefix = 'orgguardian'
    AND (PermissionsModifyAllData = true OR PermissionsViewAllData = true
         OR PermissionsAuthorApex = true OR PermissionsManageUsers = true)
  ```

  The login reads therefore ask the user who schedules the analysis for a permission that no set carries: "Monitor Login History" opens the logins of all users to that user, which is enough for the failed-login rates and the per-user breakdown; only "Manage Users" also opens `LoginGeo`, the geographic enrichment of logins, on which the by-country breakdown and the "new country" and "impossible travel" anomalies depend (the Troubleshooting Guide §4, limit 1). Likewise, the read of the registry of verification methods (`TwoFactorMethodsInfo`), on which the assessment of MFA in the toxic combination depends, requires "Manage MFA in API", which no set carries: without it, the combination is assessed without MFA and opens in `High`; granted to the scan account only, never to every user who clicks "Run scan now", it lets the combination become `Critical` when the absence of an MFA method is read (the Security FAQ §1). That permission also opens the writing of users' MFA records through the API (the `TwoFactorInfo` object): weigh it before granting it. The same holds for the OAuth register, which no set opens: the connected apps (`ConnectedApplication`), the external client apps and the tokens of all users (`OAuthToken`) are readable only by a user who holds "Customize Application". Measured on 2026-10-07 on a user holding only the shipped sets: without it, the `ConnectedApplication` read is refused, the external client apps are not visible at all, and `OAuthToken` returns only that user's own tokens; with it, the three reads succeed and uninstalled apps still holding tokens are reported. Without it, the OAuth register stops at the refused `ConnectedApplication` read, and its notice says so: the whole register stays unmeasured, its token detectors included (dormant authorizations, accounts holding live authorizations, uninstalled apps holding tokens), and the notice names the permission that reopens it. The two other token readers do not count the tokens of the account running the scan as the org's either: tokens surviving deactivated accounts are reported unread, and the configuration dependency graph marks its OAuth axis "NOT READ". "Customize Application" also allows changing the org configuration, and Salesforce grants it only together with the four permissions it depends on (`ViewSetup`, `ViewRoles`, `ManageCustomPermissions`, `ManageTranslation`, as the platform answers a permission set that lacks them): weigh it before granting it.

- **No rights on any object that does not belong to the package**, apart from three configuration objects read by the scanners (`NamedCredential`, `CspTrustedSite`, `ExternalDataSource`) — and those three are opened **Read only**, in the `OrgGuardian_Admin` set alone.
- **No overwriting of the registers**: `Attestation__c`, `Config_Baseline__c` and `Config_Baseline_Item__c` are opened with `Create + Read` by the sets that write, **never** with `Edit` or `Delete`. A row once written can no longer be rewritten or erased from the product. `Finding_Activity__c`, the log of finding handling, is `Create + Read` in `OrgGuardian_Analyst`, but `OrgGuardian_Admin` also carries `Delete` on it, for the retention purge, and therefore `Edit`, which the platform imposes together with `Delete`.
- **No user permission** in any of the five sets. The four read rights that the scan account needs (`ViewSetup`, `ViewAllUsers`, `ViewRoles`, `ViewAllProfiles`) are granted by you (§ 2, "The scan account").

What the product **writes** in your org is limited to its own records and to its own Custom Metadata Types in the `orgguardian` namespace — plus, only after an administrator's action, a Salesforce Files document (when they import a compliance mapping of their own) and the posts it publishes in the Chatter group you designate (once the Chatter alert channel is enabled). The exhaustive list is in the Security FAQ §2; the Custom Metadata Types concerned, 26 in all, are named in the Configuration Guide (§1 and §2).

---

5. Permission set and licence: two different locks

A permission set gives access to the **screen**. The licence decides whether the **feature responds**. The two are independent and combine.

- **A feature is available when its switch is on and the org's plan includes it.** Switches have a default state, on or off, which your administrator changes in **Administration ▸ Settings ▸ Feature activation**; a feature muted on the **Subscription** screen stays off whatever its switch says. Each feature also requires a plan: `FREE`, `PRO` or `ENTERPRISE`. The two conditions are cumulative: switching a feature on does not bring it into the plan, and a plan that includes a feature does not switch it on.
- **The org's plan** is the one that the publisher provisions with your subscription. Failing that, the org runs the trial of the full plan for 30 days. The trial is armed once per installation, when a user who holds **Customize Application** first opens an OrgGuardian screen (the bootstrap); it turns no feature switch on, and it is not armed if the bootstrap is done by a user without that permission. Failing that — an org without a licence, once its trial is over — it runs the `FREE` plan, the lowest: it keeps 7 days of history and includes only the features marked `FREE`, so every feature that requires `PRO` or `ENTERPRISE` stays closed there. Any paid plan includes every feature: `PRO` and `ENTERPRISE` designate the same paid level.
- **Defence against self-granting**: a `Feature_Gate__mdt` record deployed in your org can raise the plan that a feature requires, never lower it. A subscriber can **tighten** a lock, never loosen it.

Practical consequence. A user who holds `OrgGuardian_Viewer` sees the **Insights** tab; if the "In-org intelligence: baseline, anomaly, trend" feature (`Intelligence`) is off — its default — or outside the plan, the screen opens but the feature does not respond. Conversely, a paid plan shows nothing more to a user without an OrgGuardian permission set: that user does not even have the `OrgGuardian` app.

The case of amounts in euros: three layers that combine

1. **Plan and switch** — the **Cost of risk** feature (`Cost_Of_Risk`) is off by default and requires a paid plan. Unless the plan covers it **and** its switch is on, **no euro amount is calculated**, whatever the two following levers.
2. **Org setting** — **Administration ▸ Settings ▸ General settings**, setting "Visibility of monetary amounts" (key `Financials_Visibility_Mode`, stored in `Org_Setting__mdt`). Two modes: "All users" or "Permission required" (**default**).
3. **Custom permission** — `OS_View_Financials`, carried by the `OrgGuardian_Financials` set alone.

Setting mode	With `OS_View_Financials`	Without
All users	amounts visible	amounts visible
Permission required *(default)*	amounts visible	amounts **hidden**

The fallback is restrictive: if a screen cannot obtain this answer from the server, it falls back on the permission alone — amounts then appear only to holders of `OS_View_Financials`, never to every user.

---

6. Where to go next

- A more technical access-model sheet: available on request.
- What the product lets you configure without code: the Configuration Guide.
- Day-to-day operation — scans, findings, alerts, exports: the Operations Guide.
- Product vocabulary: the Glossary.
- A screen denies access: the Troubleshooting Guide.
- What the product reads, and from where: the publisher's data source matrix, available on request.
- Overall architecture: the product's architecture note, available on request.