MiaRec

System Operator Guide

MiaRec, Inc.

Table of contents

1. System Operator Guide

This guide is for the people who run the MiaRec platform itself: MiaRec staff and partner (service provider) technical teams who provision tenants, manage licenses and platform authentication, operate the ingestion and processing infrastructure, and keep the system healthy. It assumes full administrative access to the **System tenant** — the operator-owned tenant on a multitenant deployment.

Unlike the customer-facing guides, this guide freely discusses multi-tenancy, cross-tenant tools, and platform-level settings. If you administer a single organization's portal, see the Administration Guide instead; day-to-day usage is covered by the User Guide, and shared concepts by Overview & Key Concepts.

1.1 The platform at a glance

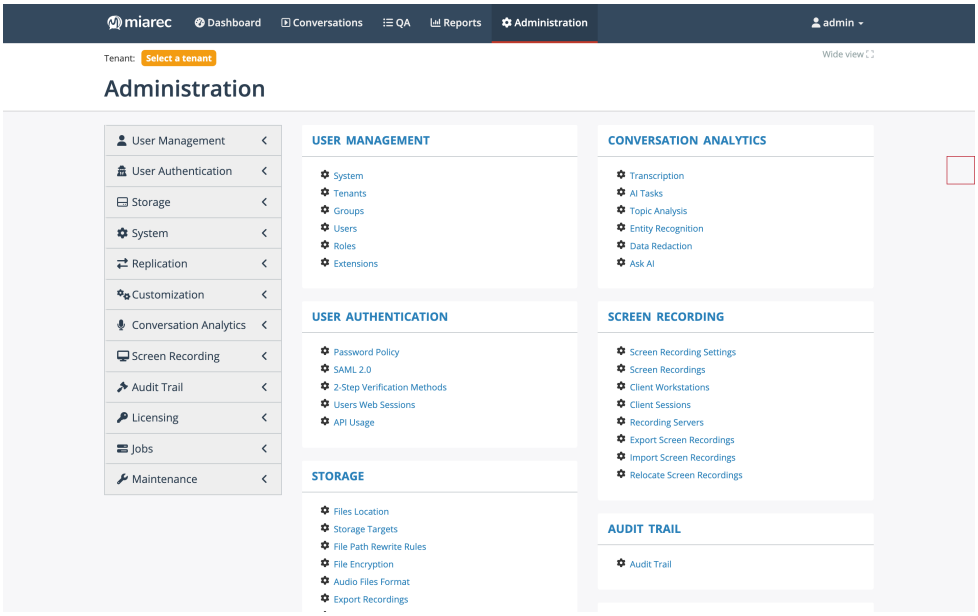
A MiaRec deployment consists of several cooperating components:

- **Web portal** — the application users and administrators sign in to; also the home of every page described in this guide (the **Administration** menu).
- **Recorder** — the capture engine that records calls from the connected phone systems. Recording is one ingestion method among several: conversations also arrive through telephony integrations (Teams, Twilio, Webex, …), third-party recorder imports, and manual or API uploads.
- **Background workers (jobs)** — the processing pipeline that transcribes conversations, runs AI tasks, redacts data, enforces retention, and synchronizes with external platforms.
- **Speech and AI engines** — configured connections to speech-to-text, generative AI, and entity recognition services used by the pipeline.

1.2 What operators do

As a system operator, working from the System tenant you:

- Create and configure tenants — isolated customer organizations — and their limits (Tenant management).
- Manage license keys, per-tenant license limits, and seat usage (Licensing).
- Run platform authentication: SAML identity providers, two-step verification methods, and system-wide password policy defaults (Platform authentication).
- Operate ingestion and recording infrastructure, storage, and replication.
- Configure and monitor the Conversation Analytics pipeline — transcription engines, AI tasks, generative AI engines — and Quality Assurance operations.
- Watch jobs, queues, system status, logs, and the cross-tenant audit trail.



The Administration home page as seen by a root operator in the System tenant. Compared with a customer administrator, the menu adds the Tenants, Extensions, Licensing, Replication, and Maintenance sections (and the System tenant context); shared sections such as User Authentication gain extra entries like SAML 2.0 and 2-Step Verification Methods.

1.3 How this guide relates to the customer guides

Many Administration pages serve both operators and customer administrators — the product does not visually separate the two audiences. This guide covers the platform side; when a setting has a customer-facing counterpart, the page links to the Administration Guide. For example, the system-tenant password policy provides the defaults that each organization's own password policy falls back to.

> **"Tenant" wording**
>
> This guide uses the term **tenant** for an isolated customer organization. The customer-facing guides deliberately never use it — they say "your organization" instead. Keep that distinction in mind when communicating with customers.

1.4 What's in this guide

- Tenant management — multi-tenancy, onboarding, tenant configuration, extensions, and view-as and impersonation.
- Licensing & packaging — the license model, license keys, and usage.
- Platform authentication — SAML identity providers, two-step verification methods, password policy defaults, portal authentication, and sessions and API usage.
- Ingestion & recording infrastructure — recording interfaces, recording rules, recorder servers, integrations, and uploads.
- Storage & data lifecycle — file locations, storage targets, encryption, limits, retention, and backup.
- Replication — multi-master asynchronous replication.
- Conversation Analytics operations — the processing pipeline, transcription, AI tasks, and engines.
- Quality Assurance operations — how Auto QA runs.
- Jobs & processing pipeline — queues and the jobs console.
- Monitoring & maintenance — system log, system status, audit trail, and troubleshooting tools.
- Platform services — email, localization, and advanced settings.
- Reporting on tenants — the cross-tenant dashboard, global reports, and tenant reports.
- Screen recording operations.

2. Tenant management

2.1 Understanding multi-tenancy

A multitenant MiaRec deployment serves multiple customer organizations — **tenants** — from a single installation. Each tenant has its own users, groups, roles, extensions, conversations, and settings, isolated from every other tenant: users of one tenant cannot see the configuration or data of another. Multi-tenancy is designed for service providers, resellers, and BPO contact centers who run conversation intelligence on behalf of other businesses — and it is how MiaRec's own cloud service is delivered.

The System tenant

One tenant is special: the **System tenant** is the operator-owned tenant your own account lives in. It is pinned as its own menu item (**Administration › User Management › System**), separate from the **Tenants** list. Working from the System tenant, with a cross-tenant role, you can:

- see and manage all tenants, their users, and their settings;
- hold platform-level settings that customer tenants fall back to (for example, the password policy defaults);
- access operator-only areas: Licensing, Replication, Jobs, Maintenance, User Authentication, and the cross-tenant dashboard.

Whether an account can see other tenants is controlled by its role's **Access scope**. Roles created in the System tenant can have the cross-tenant scopes **Unrestricted**, **System**, **All Tenants**, or **Selected Tenants**; roles inside a customer tenant are limited to their own organization. See People & Access for the general role model.

Tenant self-service

Tenants are self-service to the degree you choose. A tenant administrator can manage their own users, groups, roles, and organization-level settings — reset passwords, adjust role permissions, move users between groups — without operator involvement. Creating a tenant administrator is optional: for a hands-off customer you can create only read-only users and keep all administration on the operator

side. The Administration Guide documents everything a tenant administrator can do; settings it marks "managed by your service provider" are the ones that live in this guide.

How conversations reach the right tenant

Each tenant defines association rules — extension semantics plus matching rules — that MiaRec uses to assign captured conversations to the tenant and its users. Extensions themselves are managed centrally by the operator (see Extensions); the matching rules are part of each tenant's configuration (see Tenant configuration).

Enabling multi-tenancy

Multi-tenancy is a system-wide toggle:

1. Go to **Administration › Customization › Multitenancy**.
2. Click **Edit Configuration** and select the **Enable multitenancy** checkbox.

Once enabled, the tenant-related areas appear in the Administration menu: **Tenants** and **Extensions** under User Management, **License usage** under Licensing, and the cross-tenant dashboard for operator accounts.

> ⚠ **Warning**
>
> Multi-tenancy is a foundational deployment choice. Enable it before onboarding customers; converting a populated single-tenant system into a multitenant one is not a routine operation.

Related pages

- Onboarding a tenant — the end-to-end runbook.
- Tenant configuration — every section of the tenant page.
- View-as and impersonation — inspecting a tenant's data for support.

2.2 Onboarding a tenant

Goal: provision a new customer tenant from nothing to its first fully processed conversation — created, licensed, AI-enabled, connected to a conversation source, and verified.

Who can do this

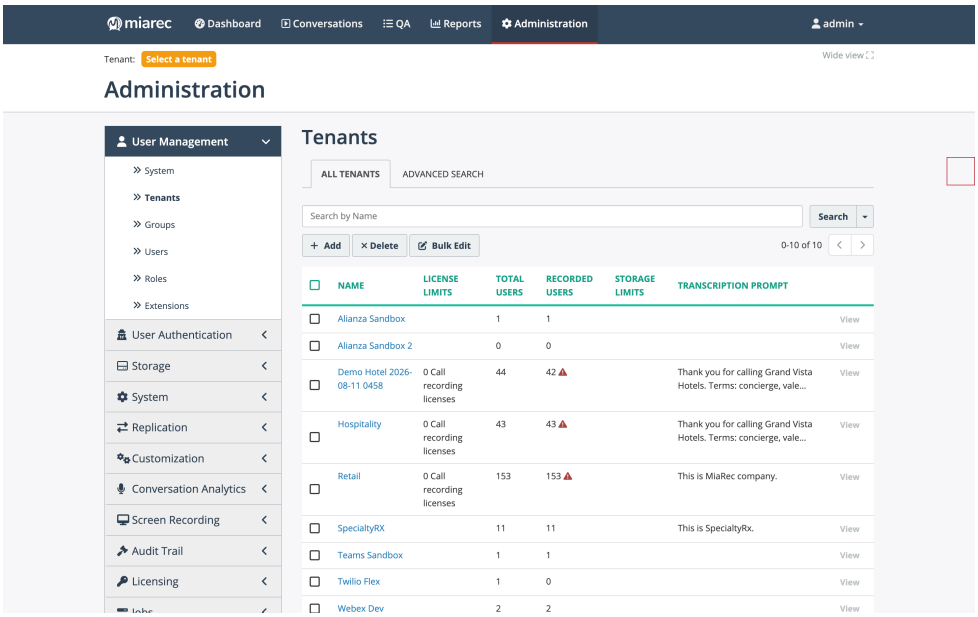
A System-tenant operator whose role can view and edit **Tenants** (and, for the verification steps, view conversations and jobs).

Before you begin

- Multi-tenancy is enabled.
- You know the customer's subscription: which license types and how many seats (see Licensing overview).
- You know how their conversations will arrive: the MiaRec recorder, a telephony integration (Teams, Twilio, Webex, …), or uploads/imports.

Step 1 — Create the tenant

1. Go to **Administration › User Management › Tenants** and click **Add**.
2. Enter the tenant **Name** and, optionally, timezone, language, and domain.
3. Save. You land on the new tenant's detail page.



The Tenants list. Each row summarizes a tenant's license limits, user counts, storage limits, and per-tenant transcription prompt; the warning icons flag tenants whose assigned seats exceed their limits.

> **Clone an existing tenant**
>
> If you provision similar customers repeatedly, open a template tenant and click **Clone Tenant** instead. Cloning copies the tenant's settings, topics, custom fields and options, scorecards (evaluation forms), AI task overrides, data redaction rules, and reports — not its users or conversations.

Step 2 — Set license limits

On the tenant's **Edit Tenant** form, in the **License limits** section, enter the seat counts the customer purchased: **Call recording**, **Screen recording**, **Live monitoring**, **Agent evaluation** (QA), and **Speech analytics** (Conversation Analytics). A type left empty means **No limits** — the tenant draws freely from the system-wide pool, so set explicit numbers for commercial customers. See License usage for how limits, keys, and per-user assignment fit together.

Step 3 — Activate AI processing

If the subscription includes Conversation Analytics:

1. Make sure the tenant has **Speech analytics** license seats (step 2) — transcription and AI tasks only run for licensed users.
2. Configure the tenant's transcription settings — engine access, per-tenant prompt, and any service limits (LLM tokens, transcription minutes) — see Transcription tenant configuration.
3. Review which global AI tasks apply to the tenant and add per-tenant overrides if the customer needs custom prompts.

Step 4 — Connect the conversation source

Depending on how conversations arrive:

- **MiaRec recorder** — in the tenant's **Recording settings**, pick the **Recording Interface** and configure the **Associate calls with tenant** rules (extension semantics, uniqueness, matching rule) so captured conversations land in this tenant. Then add the tenant's extensions (individually or as a range). See Recording interfaces.
- **Telephony integration** — connect the platform from the integration catalog and follow the platform's dedicated integration guide.
- **Uploads / imports** — review the upload settings or set up an import job.

Step 5 — Create users, or let auto-provisioning do it

Create the tenant's groups, roles, and users manually (or hand that to a tenant administrator), or enable **User auto provisioning** on the tenant so accounts are created automatically the first time an unknown extension produces a conversation. Auto-provisioning assigns each new user a group, role, recording setting, licenses, and (optionally) web portal access derived from their extension — see Tenant configuration for every field.

Step 6 — Verify the first conversation

1. In the page header, use the tenant selector (**Tenant: Select a tenant**) to view the new tenant (see View-as and impersonation).

2. Place a test call (or trigger the integration/upload) and confirm the conversation appears under **Conversations**, assigned to the expected user.
3. If AI is enabled, open the conversation and check that a transcript and AI results appear after processing completes.
4. If something is missing, check the pipeline: **Administration** › **Jobs** for the transcription and AI task runs and their logs — see the jobs console — and the per-tenant usage dashboards described in transcription monitoring.

Expected result

The tenant exists with correct license limits; conversations arrive, are associated with the tenant and its users, and (when subscribed) come back transcribed and analyzed. Users can sign in — or will be auto-provisioned as their conversations arrive.

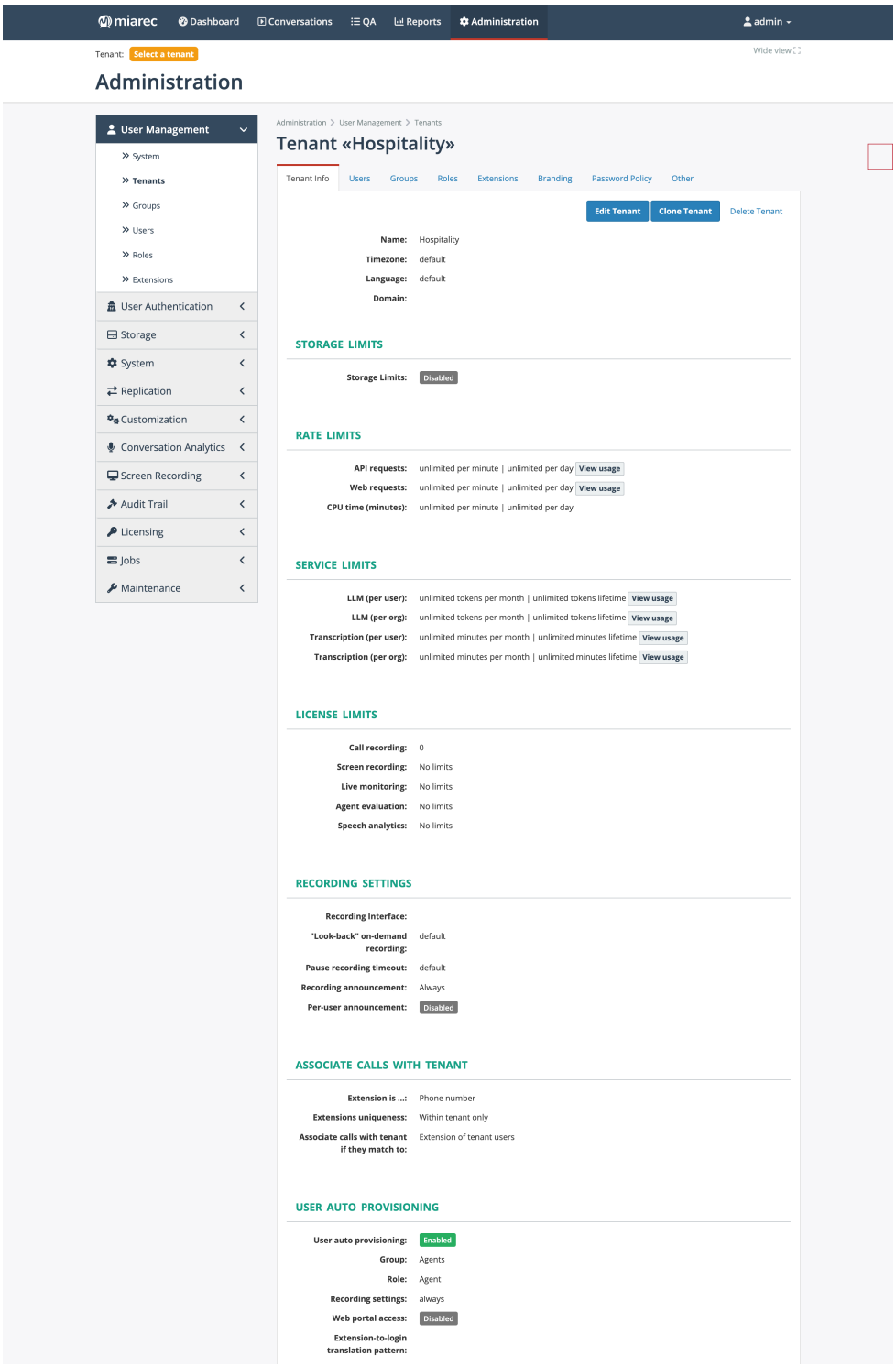
Related pages

- Tenant configuration — the full reference for every setting touched above.
- Licensing overview — the license model.
- Onboarding verification: cross-tenant dashboard — watch the tenant's volumes once live.

2.3 Tenant configuration

This page is a reference for the tenant detail page — the hub where an operator provisions everything about one tenant. Open it from **Administration › User Management › Tenants** (or **› System** for the System tenant itself).

The page is tabbed: **Tenant Info** (everything described below), plus **Users**, **Groups**, **Roles**, **Extensions**, **Branding**, **Password Policy**, and **Other** — per-tenant views of the same objects the tenant's own administrators manage. The toolbar offers **Edit Tenant**, **Clone Tenant**, and **Delete Tenant**.



The Tenant Info tab packs all per-tenant provisioning into one page: storage, rate, and service limits, license limits, recording settings, association rules, and auto-provisioning (shown above), followed by screen recording and audio settings and an inline audit trail further down.

Storage limits

Whether the tenant's audio and screen-recording storage is capped. The limits themselves and the enforcement job are documented in Storage limits; this section shows the tenant's current state (for example **Disabled** — no cap).

Rate limits

Request quotas protecting the platform from a single tenant's traffic:

- **API requests** and **Web requests** — per minute and per day.
- **CPU time (minutes)** — per minute and per day.

The API and Web rows have a **View usage** link to the tenant's request statistics — the same data as API usage. Unset values mean unlimited.

Service limits

Metered AI consumption caps, each settable per user and per organization, per month and lifetime:

- **LLM** — generative AI tokens consumed by AI tasks.
- **Transcription** — speech-to-text minutes.

View usage links open the corresponding usage dashboards. See transcription tenant configuration for how these limits interact with pooled limits and user licensing.

License limits

Per-tenant seat caps for the five license types: **Call recording**, **Screen recording**, **Live monitoring**, **Agent evaluation**, and **Speech analytics**. A type with no value shows **No limits** — the tenant is in dynamic mode and draws from the system-wide pool. This is one of the three places licensing lives (keys, limits, assignment) — see License usage.

Recording settings

Capture behavior for conversations recorded by the MiaRec recorder:

- **Recording Interface** — which phone-system integration records this tenant's calls (see Recording interfaces).
- **"Look-back" on-demand recording** — whether on-demand recordings include audio from before the trigger.
- **Pause recording timeout** — automatic resume after a pause.
- **Recording announcement** and **Per-user announcement** — announcement behavior, when the announcement feature is enabled on the system (see Announcement rules).

Values shown as *default* fall back to the system-wide setting.

Associate calls with tenant

The rules that route captured conversations into this tenant and to its users:

- **Extension is …** — what an extension means for this tenant: **Phone number**, **Last digits**, **Phone name**, **Phone ID**, **Phone number or name**, **Broadworks user ID**, **Metaswitch extension**, **Agent ID**, or **Legacy (Phone number, name or BW user ID)**.
- **Extension uniqueness** — **System wide** or **Within tenant only** (allow the same extension to exist in different tenants).
- **Associate calls with tenant if they match to** — the tenant-matching rule: **Extension of tenant users**, **Broadworks Service Provider + Group**, **MetaSwitch System + Group**, **Cisco Partition**, **SIP URI host part**, or an **Advanced condition** expression.
- **Record unknown users** — whether conversations matching the tenant but not any user are recorded (**Record**) or dropped (**Do not record**). Recording unknown users is a prerequisite for auto-provisioning.

User auto provisioning

When enabled, MiaRec creates a user account automatically the first time a conversation arrives for an unknown extension that matches the tenant. Configure what each auto-created account gets:

- **Group** and **Role**;
- **Recording setting** — **Always** or **On-demand**;
- **Web portal access** — whether the account can sign in, with an **Extension-to-login translation pattern** and **rule** (a regular expression deriving the login from the extension) and the **Authentication type** for the new account;
- **Provisioned licenses** — which license seats each auto-created user receives (call recording, screen recording, live monitoring, agent evaluation, speech analytics).

Auto provisioning is allowed only when the association rules and **Record unknown users** are configured accordingly (the form enforces this).

Screen recording

Enables screen capture for the tenant and manages the screen-recording token used by client workstations. See Screen recording operations.

Audio settings

Per-tenant overrides of the audio storage format: **File format**, **Audio format**, **AGC filter**, and **PLC filter** — *default* means the system-wide values from Audio format apply.

Per-tenant transcription settings

The Tenants list includes a **Transcription Prompt** column showing each tenant's custom transcription prompt (vocabulary hints passed to the speech engine). These settings are edited from the tenant's transcription settings — see Transcription tenant configuration.

Audit trail

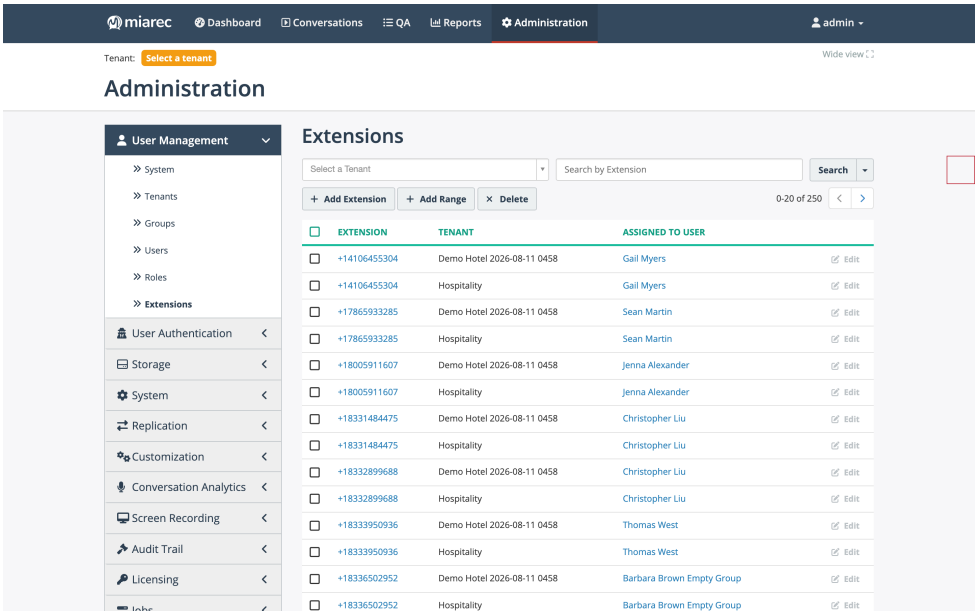
The bottom of the tenant page embeds the audit trail filtered to this tenant — handy when checking who changed a limit or setting. The full cross-tenant view is at Audit trail.

Related pages

- Onboarding a tenant — the order to configure these sections for a new customer.
- Extensions — managing the extension pool the association rules match against.
- Licensing overview — what each license type gates.

2.4 Extensions

An **extension** is the telephony identifier (a phone number, phone name, Broadworks user ID, agent ID, …) that links captured conversations to a user. On a multitenant system, the extension pool is managed centrally by the operator at **Administration › User Management › Extensions** — a flat, cross-tenant list.



The Extensions list spans all tenants. Each row shows the extension, its tenant, and the user it is assigned to; the same number can legitimately appear in several tenants when their extension uniqueness is "Within tenant only".

What "extension" means per tenant

The *semantics* of an extension are defined per tenant in its association rules — **Extension is …** (Phone number, Last digits, Phone name, Broadworks user ID, Agent ID, …) and **Extension uniqueness** (system-wide, or unique within the tenant only). See Tenant configuration. Choose the semantics that match the customer's phone platform — for example, Broadworks user IDs on a Broadworks platform, phone numbers elsewhere, phone names when several users share a number and only the name part is unique.

Managing the list

- **Add Extension** — create a single extension: enter the value and the owning **Tenant**.

- **Add Range** — create a numeric range at once: **Begin Range**, **End Range**, and the **Tenant**.
- **Delete** — remove selected extensions.
- Filter with the **Select a Tenant** drop-down or search by extension value.

Assigning extensions to individual users happens on the user account (or automatically via user auto-provisioning). Tenant administrators can re-allocate the extensions you have granted their tenant among their own users, but the cross-tenant pool itself — which extensions exist and which tenant owns them — is operator-controlled. The customer-side view of associating conversations with users is documented in Associating conversations with users.

Related pages

- Tenant configuration — extension semantics and association rules.
- Onboarding a tenant — where extension setup fits in provisioning.
- Phone number normalization — rewriting numbers before they are matched.

2.5 View-as and impersonation

Two tools let you see the platform the way a customer does — an everyday necessity for support and onboarding verification. The **tenant selector** scopes your own session to one tenant's data; **impersonation** signs you in as a specific user.

Viewing a tenant's data (tenant selector)

Operator accounts that can view tenants get a tenant selector in the page header, next to the page title: **Tenant:** followed by the selected tenant's name — or an orange **Select a tenant** badge when none is selected. Click it, pick a tenant, and the data pages — Dashboard, Conversations, QA, Reports — show that tenant's data, as its administrators see it. The selection sticks for your session until you change it.

Use the tenant selector to:

- verify a new tenant's first conversations during onboarding;
- reproduce what a customer reports seeing on their dashboards or conversation lists;
- work with tenant-scoped pages (for example, uploading a conversation into a specific tenant).

The cross-tenant dashboard offers the complementary top-down path: an aggregate view of all tenants with drill-down into any tenant's analytics.

> **Note**
>
> The tenant selector changes which data *you* are viewing with your own operator permissions. It does not reproduce a particular user's role, permissions, or field-visibility settings — for that, impersonate the user.

Impersonating a user

Impersonation turns your session into that user's session — same role, permissions, access scope, and profile — so you can see exactly what they see and reproduce their issue.

1. Go to **Administration › User Management › Users** (or the tenant's **Users** tab) and open the user.
2. Click **Impersonate** in the toolbar.
3. You are now signed in as that user. When finished, open the user menu (your name, top right) and click **Exit user impersonation** to return to your own session.

Notes and safeguards:

- Impersonation requires the **Impersonate** permission on your role, and the target user must be visible to your access scope.
- You cannot impersonate your own account. Root and system-administrator accounts cannot be impersonated at all, nor can inactive users or users without web-portal access.
- Every impersonation is recorded in the audit trail as a sign-in of the target user annotated with the impersonator's name and login — customers can see that support accessed the account, and by whom.

> ⚠ **Warning**
>
> While impersonating, anything you do — playing conversations, editing settings, deleting data — happens as the impersonated user. Keep support sessions read-only unless the customer has asked for a change.

Related pages

- Sessions and API usage — reviewing and terminating users' web sessions.
- Audit trail — the record of impersonation and other actions.

3. Licensing & packaging

3.1 Licensing overview

MiaRec capabilities are licensed per user as **seats**. This page explains the license model and — because licensing information is spread across three places in the product — where each piece lives.

Seat types

Five license types are assignable to users. The UI shows each with a short code:

UI label	Code	What it entitles
Call recording license	REC	The user's conversations are captured by the recorder
Screen recording license	SCR	The user's desktop is captured during conversations
Live monitoring license	MON	The user's active conversations can be listened to in real time
Agent evaluation license	EVAL	The user's conversations can be evaluated manually (QA)
Speech analytics license	SPCH	The user's conversations are transcribed and analyzed by AI

Label renames in progress

The display names "Agent evaluation license" and "Speech analytics license" correspond to the **Quality Assurance** and **Conversation Analytics** products. This guide uses the current UI labels in procedures; the codes (REC/MON/EVAL/SPCH/SCR) are identifiers and do not change.

Besides the five seat types, the recorder license can carry **recording session** and **monitoring session** entitlements — concurrency pools counted per simultaneous conversation rather than per user. They are not assignable to users.

Where licensing is enforced

A seat gates its stage of the pipeline: capture requires a recording seat; transcription and AI tasks run only for users with a speech analytics license; manual evaluations require the evaluated user to hold an agent evaluation license; live monitoring requires a monitoring seat; screen capture requires a screen

recording seat. Auto QA is driven by the AI pipeline and therefore follows the speech analytics license, not the evaluation license.

Licenses and packaging

Commercially, customers subscribe to the MiaRec products — Conversation Analytics and Quality Assurance (see Subscriptions & Licensing for the customer-facing framing). In the product today there is no package-level switch: a subscription is realized as a combination of license seats, role permissions, and feature flags. When you onboard a customer, you translate their subscription into per-tenant license limits and, where applicable, flags — see Onboarding a tenant.

The three places licensing lives

What	Where	Documented in
License keys — the system-wide entitlement pools issued to your recorder and screen-recording modules	**Administration › Licensing › Recording License / Screen Recording License**	License keys
Per-tenant limits — how many seats of each type a tenant may consume	Tenant page › **License limits**	Tenant configuration
Per-user assignment — which users hold which seats	Each user account (or auto-provisioning)	Customer side: User accounts

Usage across all three is summarized on the **License usage** page — see License usage. A tenant with no limit set for a type shows **No limits** and draws dynamically from the system-wide pool.

Related pages

- License keys — keys, expiration, and module status.
- License usage — per-tenant and per-group seat consumption.

3.2 License keys

License keys carry the system-wide entitlements — how many seats of each type the deployment as a whole may use, and until when. Keys live under **Administration › Licensing**, separate from the per-tenant limits and per-user assignment described in License usage.

Recording license

Administration › Licensing › Recording License shows the license of each registered recorder. Open a recorder's license to see:

- the licensed count for each type — recording seats/sessions, monitoring seats/sessions, evaluation seats, speech analytics, screen recording seats — alongside the number currently **required** by assignments;
- **License Expiration** and **Maintenance Expiration**, with the remaining (or overdue) time;
- a **View license utilization** link — per-recorder usage broken down by tenant and user.

Click **Edit License** to install or replace the license key.

> **License info requires a running recorder**
>
> The license details are reported by the recorder service. If the recorder is offline, the page shows only *"License info is not available because the server is offline for …"* — bring the recorder back online (see Recorder servers) to read or verify the license.

Screen recording license

Administration › Licensing › Screen Recording License lists the screen-recording module instances with their **Instance**, **Location**, **Version**, and **Status**, and holds the license for the screen-recording capability. See Screen recording operations for the surrounding infrastructure.

When a key changes

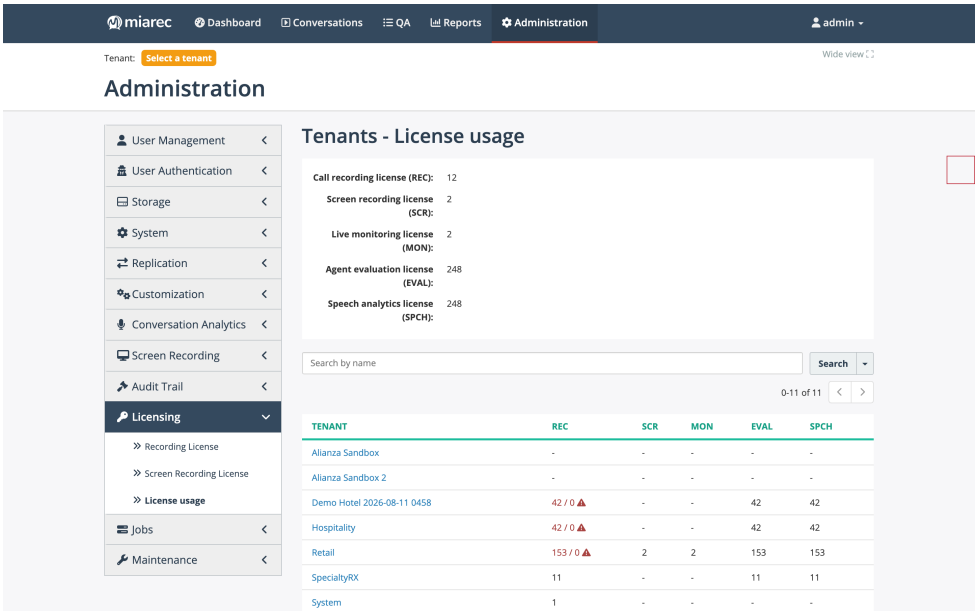
Installing a new key changes the system-wide pools immediately. If the new key is smaller than the seats already assigned, the affected types show license warnings on the License usage page — resolve them by unassigning seats or correcting the key.

Related pages

- Licensing overview — the license model and seat types.
- License usage — who is consuming the licensed seats.

3.3 License usage

Administration › Licensing › License usage is the operator's consumption view: how many seats of each type are in use across the system, tenant by tenant. Use it to spot tenants exceeding their limits, reconcile against contracts, and find where a "no license available" complaint comes from.



The License usage page. The header totals the seats in use per type (REC/SCR/MON/EVAL/SPCH); the table breaks them down per tenant. "42 / 0" with a warning icon means 42 seats are assigned against a limit of 0 — a license issue to resolve.

Reading the page

- The header lists the total seats currently in use for each of the five types — **Call recording license (REC)**, **Screen recording license (SCR)**, **Live monitoring license (MON)**, **Agent evaluation license (EVAL)**, **Speech analytics license (SPCH)**.
- The table shows each tenant's usage per type. A dash means no seats of that type are assigned. When a tenant's assignments exceed its limit, the cell shows *assigned / limit* with a warning icon.
- Click a tenant to open its license view: the tenant's **License limits** next to its actual usage, with **Groups** and **Users** tabs breaking consumption down further — the fastest way to find exactly which users hold a seat type.

Fixing a license issue

A warning on this page means assignments exceed the tenant's limit (or the system pool). Either raise the tenant's limit (**Edit Tenant › License limits**, if the contract allows), or reduce assignments — unassign the seat from users who do not need it, on their user accounts or in bulk. Tenant administrators can re-distribute their organization's seats themselves; the limits are yours.

Where limits, keys, and assignment live

Licensing spans three places — this page is the reconciliation view across all of them:

Piece	Where to change it
System-wide pools (key)	License keys
Per-tenant limits	Tenant page › License limits
Per-user assignment	The user's account (User accounts); defaults for new users via auto-provisioning

License columns in reports

License data is also available as report columns for scheduled reconciliation:

- **User license columns** (in Calls Summary and User Details report types): *User - License - Call Recording*, … - *Screen Recording*, … - *Live Monitoring*, … - *Agent Evaluation*, … - *Conversation Analytics* — the number of seats of that type assigned to each user. (The report column already uses the "Conversation Analytics" name even though the license field label is still "Speech analytics license".)
- **Tenant reports** include per-tenant license columns (*License* - …) alongside storage and usage columns — see Tenant reports.

Related pages

- Licensing overview — the seat types and what they gate.
- License keys — the system-wide pools.

4. Platform authentication

4.1 SAML 2.0 identity providers

Registering a SAML 2.0 identity provider (IdP) lets a customer's users sign in to MiaRec with their own identity system — Google, Microsoft Entra ID, Okta, or any SAML 2.0 provider. IdP registration is a platform-level (System tenant) task; each organization's SSO *policy* (enabled, mandatory, restricted domains) is customer-side — see Single sign-on in the Administration Guide.

IdPs are managed at **Administration › User Authentication › SAML 2.0** — a list of registered providers with their **Name**, **Status** (Active / Not active), and **Application domain**, plus **Add** and **Delete**.

> **Tip**
>
> For Google, Microsoft, and Webex sign-in, the integration catalog (**Administration › System › Integrations**) offers dedicated SSO connections that are simpler to set up than a generic SAML registration.

Registering an identity provider

Click **Add** and fill in:

- **Status** — **Active** to enable the provider for sign-ins.
- **Name** — shown to users on the sign-in page ("Sign in with *Name*").
- **Application domain** — the domain users type to reach the portal (without `https://`). MiaRec supports multiple IdPs side by side: give each customer (or group of tenants) its own subdomain — `customer1.example.com`, `customer2.example.com` — and register one IdP per domain.
- **SAML Login URL** — the IdP's Single Sign-On service URL (HTTP-Redirect binding).
- **SAML Logout URL** *(optional)* — the IdP's Single Log-Out URL; when set, signing out of MiaRec also signs the user out of the IdP.
- **Identity Provider X.509 certificate (PEM format)** — paste the certificate from the IdP's metadata (the content of the `X509Certificate` tag, without the `BEGIN/END CERTIFICATE` lines).
- **Login attribute** — the assertion attribute MiaRec matches against the user's login (default `uid`; typically set to `email`).

After saving, the provider's view page shows the values the IdP side needs — the **service provider** half of the handshake:

- **SP Entity ID** and **SP ACS URL** (Assertion Consumer Service URL) — enter these when creating the MiaRec app in the identity provider;
- **SP Metadata URL** and a **Download Service Provider's metadata file** link, for IdPs that accept metadata import.

Testing the configuration

On the provider's view page, click **Test Single Sign-On**. MiaRec sends an authentication request to the IdP and displays the raw response, including the **assertion attributes** the IdP returned. Confirm the attribute named in **Login attribute** (for example `email`) is present — if not, fix the attribute mapping on the IdP side.

Enabling SAML for users

A user signs in via SAML when their account's **Authentication type** is **Single Sign-On** — set it on the user form, or with **Bulk Edit** for many users at once (customer administrators can do this too; see User

accounts). The user's **Login** must match the value the IdP sends in the login attribute. On first SSO sign-in, MiaRec links the IdP identity to the account; the linking rules are described in Single sign-on.

Related pages

- Password policy — system defaults, including the platform-wide SSO status.
- Two-step verification methods — organizations can exempt SSO users from 2FA.
- Portal authentication — the other external authentication options.

4.2 Two-step verification methods

Two-step verification (2FA) asks users for a one-time code in addition to their password. The *methods* users can verify with are enabled platform-wide here, in the System tenant; whether 2FA is *required* is decided per organization or per user on the customer side (see Two-step verification in the Administration Guide). Until at least one method is enabled below, customer portals show *"2-step verification is not configured on this system"* and enforcement settings have no effect.

Go to **Administration › User Authentication › 2-Step Verification Methods**. The page lists the three methods, each with its status and a **Configure** link:

- **Time-based one-time password verification** (TOTP) — authenticator apps.
- **SMS-based verification** — codes sent by text message via Twilio.
- **Email-based verification** — codes sent by email.

Users enroll their methods themselves from **My Profile › Security** (see Account security in the User Guide).

TOTP (authenticator app)

The recommended method — no external service or per-message cost.

1. Click **Configure** next to **TOTP verification**.
2. Select **Enable time-based one-time password verification**.
3. Use the **Test application** section to verify: scan the QR code (or enter the secret key) in an authenticator app — Microsoft Authenticator, Google Authenticator, Authy, and similar — and confirm a generated code with **Verify Code**.
4. **Save**.

SMS via Twilio

SMS codes are sent through a Twilio account you provide. You need an upgraded (paid) Twilio project with an SMS-capable phone number; for throughput above 1 SMS per second, combine several numbers into a Twilio Messaging Service.

1. Click **Configure** next to **SMS-based verification** and select **Enable**.
2. Enter the **Twilio Account SID** and **Twilio Auth Token** from the Twilio console's project info.
3. Set **Twilio phone number or Messaging Service ID** — a Twilio number you own in E.164 format (for example `+16175551212`), a short code, or a Messaging Service SID.
4. Choose the **Verification code length** (6–8 digits) and optionally edit the **Text message** template sent to users.
5. Use **Test connection settings** — enter a country code and phone number and click **Test a Connection**; confirm the test SMS arrives.
6. **Save**.

> **Note**
>
> Twilio charges per message and per phone number — see Twilio's SMS pricing. Protect the Twilio account itself with 2FA on the Twilio side.

Email

Email codes are sent through the platform's SMTP integration, so configure email first — the page warns *"SMTP is not configured"* (or *"Warning! Webportal URL is not configured."* — the URL is set in Advanced settings) until the prerequisites are in place.

1. Click **Configure** next to **Email-based verification** and select **Enable**.
2. Choose the **Verification code length** (6–8 digits).
3. Use **Test connection settings** — enter an email address and click **Test a Connection**; confirm the code email arrives.
4. **Save**.

The wording of the code emails comes from the **2-step verification sign in** template (and its setup counterpart) in the email template catalog — see Email.

After enabling

Two-step verification is considered "on" as soon as at least one method is enabled. Organizations can then require it for all users or non-SSO users, and individual accounts can be flagged to require it — both are customer-side settings documented in Two-step verification.

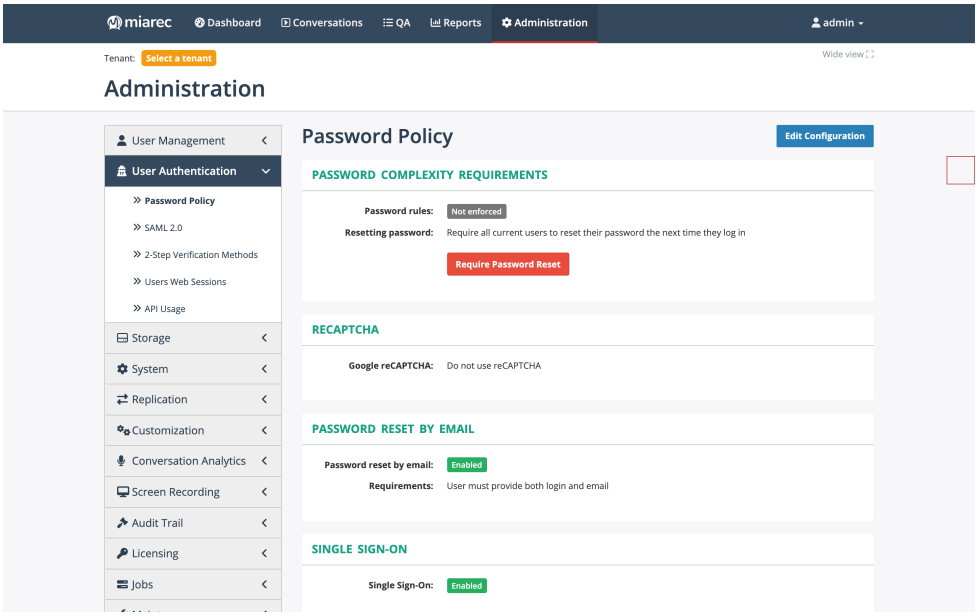
Related pages

- Password policy — where the system-default 2FA enforcement lives.
- Email — SMTP configuration and email templates.

4.3 Password policy (system defaults)

The System tenant's **Password Policy** page holds the platform-wide defaults: whenever a customer organization has not set its own rule, the value configured here applies (their page shows *"Not enforced (using a system default setting)"* or *"System default"*). The same page also contains the system-level reCAPTCHA setting, which exists only here.

Go to **Administration › User Authentication › Password Policy** while in the System tenant, and click **Edit Configuration** to change any section. The per-organization counterpart is documented in Password policy in the Administration Guide.



The Password Policy page in the System tenant. Unlike the per-organization page, it includes the Recaptcha section, and its values serve as the defaults that organizations fall back to.

Password complexity requirements

Password rules — when enforced, set the **Minimum password length** and the required character classes (uppercase, lowercase, numeric, special characters). Rules apply when passwords are created or changed; use **Require Password Reset** to force all current password-authenticated users to reset at next sign-in and bring existing accounts under new rules.

reCAPTCHA

Google reCAPTCHA protects the sign-in page against automated credential attacks. Options:

- **Do not use reCAPTCHA** (default);
- **Always use reCAPTCHA** — every sign-in includes the challenge;
- **Use reCAPTCHA to allow users to bypass failed login attempts rate limit** — the challenge appears only after repeated failures, letting a legitimate user proceed where the rate limit would otherwise block them.

Enabling it requires a Google reCAPTCHA **site key** and **secret key**.

> **Note**
>
> reCAPTCHA is rarely used in practice — failed sign-ins are already rate-limited — and the setting may be retired in a future release. Leave it at **Do not use reCAPTCHA** unless a deployment has a specific requirement.

Password reset by email

The default for the self-service **Forgot your password?** link: enabled (with the choice of requiring email only, or both login and email), or disabled. Organizations may override this; self-service reset requires SMTP to be configured.

Single Sign-On and two-factor authentication

The remaining sections mirror the per-organization page and act as system defaults:

- **Single Sign-On** — Enabled / Disabled / Mandatory, with optional restricted email domains. Identity providers themselves are registered on the SAML 2.0 page.
- **2-step verification** — Not enforced, Required for all users, or Required for non-SSO users. The verification methods must first be enabled under Two-step verification methods.

Related pages

- Password policy (per organization) — what customer administrators see and can override.

- SAML 2.0 identity providers — registering IdPs.
- Two-step verification methods — enabling TOTP, SMS, email.

4.4 Portal authentication (Metaswitch, Broadworks)

On deployments for Metaswitch or Broadworks service providers, users can sign in to MiaRec with the credentials of their carrier portal account: MiaRec forwards the submitted login and password to the portal for verification instead of checking a local password. This avoids separate MiaRec passwords for large carrier-managed user bases.

> **Flag-gated features**
>
> These options are hidden by default. They appear under **Administration › User Authentication** only when the corresponding feature flag is enabled — *Metaswitch integration* or *Broadworks integration* — see Advanced settings. Enable them only on deployments for the respective platform.

Metaswitch CommPortal

Administration › User Authentication › Metaswitch CommPortal, then **Edit Configuration**:

- **Enable Metaswitch CommPortal based user authentication**;
- **CommPortal URL** — the customer-facing CommPortal address;
- **Test connection settings** — a CommPortal **username** and **password** used only for testing; click **Test a Connection** to verify MiaRec can authenticate against the portal.

Broadworks Web Portal

Administration › User Authentication › Broadworks Web Portal, then **Edit Configuration**:

- **Enable Broadworks Web Portal based user authentication**;
- **Broadworks Web Portal URL** — in the format `https://portal.example.com`;
- **Test connection settings** — a Web Portal **username** and **password** for testing, with **Test a Connection**.

Enabling portal sign-in for users

Once a portal integration is enabled, the corresponding option — **Metaswitch CommPortal** or **Broadworks Web Portal** — appears in the **Authenticate with** choices on user accounts (and in the **Authentication type** of user auto-provisioning). Set it per user or via **Bulk Edit**; the user's MiaRec **Login** must match their portal account. Portal sign-ins are password-based from MiaRec's point of view, so two-step verification can apply to them like regular password sign-ins.

Related pages

- SAML 2.0 identity providers — the standards-based alternative for SSO.
- Advanced settings — the feature flags that reveal these pages.

4.5 Sessions and API usage

Two read-mostly pages under **Administration › User Authentication** give operators a cross-tenant view of portal activity: **Users Web Sessions** (who is signed in right now) and **API Usage** (request volumes against each tenant's limits). Customer administrators have the same pages scoped to their own organization — see Web sessions and API usage in the Administration Guide.

Users Web Sessions

Administration › User Authentication › Users Web Sessions lists every active session across all tenants. Columns: **Tenant**, **User**, **Login**, **IP address**, and **Session Start Time**, with a **View** action for session details. Narrow the list with the **Select a Tenant** drop-down or search by user, login, or IP address.

Typical operator uses:

- security review — spot sign-ins from unexpected addresses, or accounts that should be disabled;
- support — confirm whether a user is currently signed in, and from where;
- integration hygiene — automation accounts that sign in per request instead of keeping a session show up as long runs of near-identical rows.

Select sessions and click **Terminate** to sign the users out immediately. Termination does not disable the account — pair it with a password reset or disabling web access on the user account when responding to a compromise.

API usage

Administration › User Authentication › API Usage shows request statistics per tenant: pick the tenant, and the page displays its **API access limits** and **Web access limits**, plus historical counters for the selected date range — API and web requests **served** and **denied** — with a per-day chart.

A *denied* request was rejected by the tenant's rate limits. Sustained denials usually mean a customer integration is polling too aggressively; either the integration slows down, or you raise the tenant's limits in its rate limits (the tenant page's **View usage** links land on this same data).

Access to the REST API is permission-controlled per role, and the API itself is documented in the REST API Developer Guide.

Related pages

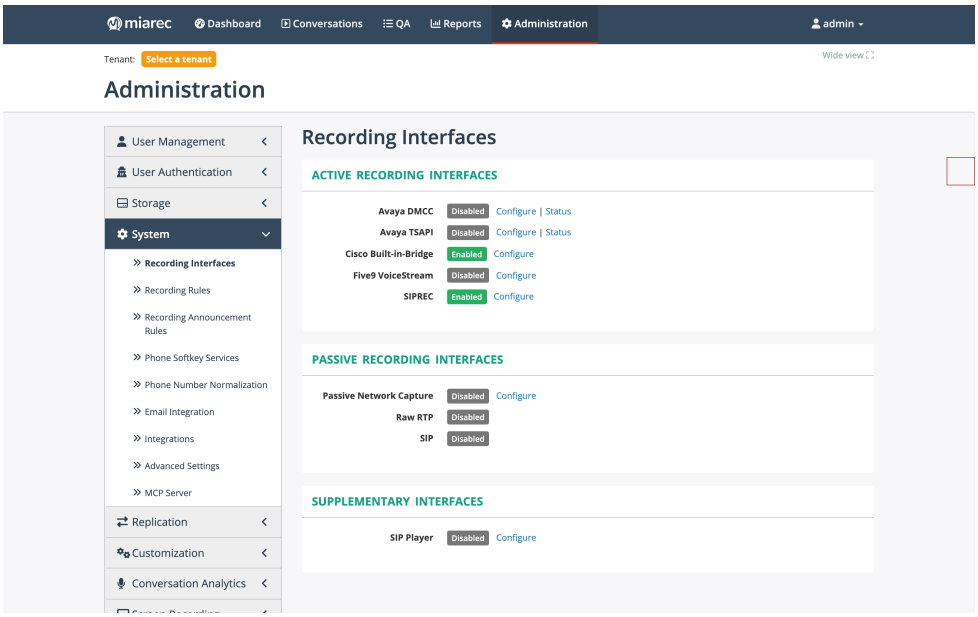
- Tenant configuration — rate limits — where the enforced limits are set.
- View-as and impersonation — acting on what you find here.
- Audit trail — the historical record of sign-ins and actions.

5. Ingestion & recording infrastructure

5.1 Recording interfaces

A recording interface is the protocol the MiaRec recorder uses to capture audio from a phone system — SIPREC from a session border controller, Cisco Built-in-Bridge from a Cisco call manager, a passive network tap, and so on. This page covers where interfaces are enabled and configured; the recording decision itself (record or not) is made by recording rules.

Where: Administration › System › Recording Interfaces. The page groups the available interfaces and shows an **Enabled**/**Disabled** badge with a **Configure** link for each.



The Recording Interfaces page. Notice the three groups — active, passive, and supplementary — and the per-interface Enabled/Disabled badges; the Avaya interfaces additionally expose a Status link.

Interface groups

- **Active recording interfaces** — the phone system delivers media to the recorder deliberately: **Avaya DMCC**, **Avaya TSAPI**, **Cisco Built-in-Bridge**, **Five9 VoiceStream**, and **SIPREC**. SIPREC is the general-purpose interface used by session border controllers and by the Microsoft Teams integration (the MiaRec TeamsBot sends SIPREC INVITEs to the recorder).

- **Passive recording interfaces** — the recorder captures traffic from a network tap or mirror port: **Passive Network Capture**, **Raw RTP**, and **SIP**.
- **Supplementary interfaces** — **SIP Player**, used for audio playback (for example, live monitoring by phone) rather than capture. **Cisco TAPI** also appears in this group when its vendor flag is enabled (see below).

The **Avaya DMCC** and **Avaya TSAPI** interfaces each provide a **Status** page with the registered device lists, so you can verify which stations are being observed.

Vendor flags

Some interfaces and their related menus are gated by feature flags in **Administration › System › Advanced Settings**:

- **Cisco TAPI** appears in the interface list only when the Cisco TAPI integration flag is allowed.
- **Five9 VoiceStream** is always listed here, but the dedicated Five9 menu pages (settings, authorizations, subscriptions) appear only when the Five9 integration flag is allowed.

Scope of the settings

Interface settings configured on this page are stored as base recorder settings and apply to **all** recorder servers in the deployment. In multi-platform deployments (for example, BroadWorks plus Microsoft Teams), you can override individual keys per recorder under **Administration › Maintenance › Recording Servers** — see Recorder servers.

Phone systems that are integrated through API-based synchronization rather than the recorder — such as Twilio or Webex Native Recording — do not use a recording interface; they are connected through the Integrations catalog.

Related pages

- Recording rules — deciding which calls are recorded.
- Recorder servers — the recorder registry and per-recorder overrides.
- Integrations — API-based telephony integrations.
- Microsoft Teams Integration Guide — an example of a SIPREC-based integration end to end.

5.2 Recording rules

Recording rules decide, per call, whether the recorder keeps the audio. They let you record everything for one tenant, record on demand for another, and skip internal calls for a third — without touching the phone-system side of the integration.

Where: Administration › System › Recording Rules. The page has four tabs: **Settings** (the default rule and related options), **Custom rules [before]**, **Auto rules**, and **Custom rules [after]**.

> **Note**
>
> This page describes the generic rule engine. For a worked, platform-specific walkthrough — including the platform-side policy that must deliver calls to the recorder in the first place — see Teams Recording Rules in the Microsoft Teams Integration Guide.

Rule levels and evaluation order

MiaRec evaluates rules from most to least specific. The first matching rule wins:

1. **Custom rules [before]** — exception rules you create, evaluated first.
2. **User auto rules** — from the **Record** setting on each user profile (**Always** / **On-demand** / **Never** / **Default**); applies to calls associated with that user.
3. **Tenant auto rules** — from the **Record unknown users** setting on the tenant profile; applies to calls associated with the tenant but not with any user.
4. **Custom rules [after]** — rarely used; evaluated after the auto rules.
5. **Default recording rule** — the system-wide fallback for calls that matched nothing above.

The user- and tenant-level settings are edited on the user and tenant profiles; the **Auto rules** tab shows the resulting rules in one place for convenience. Correct rule matching therefore depends on calls being associated with the right tenant and user — see the association settings in Tenant configuration.

The default rule and settings

On the **Settings** tab, click **Edit Configuration**:

- **Default recording rule** — one of:
  - **Record** — record the call.
  - **On-demand** — the user can turn recording on or off during the call; if no action is taken, the recording is discarded.
  - **On-demand (default: keep)** — same, but the recording is kept when no action is taken.
  - **Do not record** — never record.
- **Ignored calls processing** — "Save metadata for ignored calls": keeps the call record (without audio) for calls the rules declined.
- **On-demand recording** — "'Look-back" on-demand recording': when enabled, on-demand recordings include audio from the beginning of the call; otherwise audio starts at the moment recording is triggered.
- **Pause recording timeout** — automatically resume a paused recording after the given number of seconds (0 disables the timeout).

Custom rules

Custom rules express exceptions: *record everything for this tenant unless the call is outbound*, or *do not record calls of a given type*. Click **Add** on the **Custom rules [before]** tab and define:

- **Tenant** — the tenant the rule belongs to.
- **Condition** — the match criteria (call attributes such as direction or platform-specific metadata fields).
- **Action** — **Record** / **On-demand** / **On-demand (default: keep)** / **Do not record**.
- **Position** — before or after the auto rules (before is the common case).

> ⚠ **Prefer 'Do not record' exceptions**
>
> Structure custom rules as *record by default, except…* with the **Do not record** action. A custom rule with the **Record** action can consume an additional recording license when it triggers on a call for a user who is already licensed through the auto rules, leading to license oversubscription.

Early check of recording filters

By default, the recorder postpones rule evaluation until media (RTP) is received: recording starts for every call and the audio is discarded later if the rules decline it. That is invisible to callers — unless recording announcements are enabled, in which case participants can hear an announcement for a call that is ultimately not recorded.

To evaluate rules before recording starts, enable **Early check of recording filters** under **Administration › System › Recording Interfaces › SIPREC** (Recording Filters Behavior section). In multi-platform deployments, apply it per recorder instead with an override (**Category** `Protocol::SIPREC`, **Key** `EarlyCheckRecordingFilters`, **Value** `true`) — see Recorder servers.

Related pages

- Announcement rules — notifying participants that a call is recorded.
- Recording settings in the Administration Guide — the per-user recording settings as customer administrators see them.
- Tenant configuration — tenant association rules and per-tenant recording settings.

5.3 Announcement rules

Recording announcement rules control whether call participants are notified that the call is being recorded. They operate on top of recording rules: recording rules decide whether a call is recorded; announcement rules decide whether participants are told about it. If the recording rules decline a call, no announcement is played.

Where: Administration › System › Recording Announcement Rules — the same four-tab structure as recording rules: **Settings**, **Custom rules [before]**, **Auto rules**, **Custom rules [after]**.

> **Flag-gated feature**
>
> The menu item and the per-tenant announcement settings appear only after you select **Allow** for the recording announcement configuration in **Administration › System › Advanced Settings**.

How the announcement is delivered depends on the phone platform. For example, with Microsoft Teams, MiaRec only triggers the notification through Teams APIs — the recording banner in the Teams client and the audio message played to PSTN participants are controlled by Microsoft Teams itself. See Teams Recording Announcement for that platform's specifics.

Rule levels and evaluation order

Announcement rules mirror the recording-rule hierarchy. The first match wins:

1. **Custom rules [before]** — exception rules, evaluated first.
2. **User auto rules** — from the **Recording announcement** setting on the user profile (**Always** / **Never** / **Default**).
3. **Tenant auto rules** — from the **Recording announcement** setting on the tenant profile (**Always** / **Never** / **Default**).
4. **Custom rules [after]** — rare.
5. **Default rule** — the system-wide fallback: **Announce** or **Do not announce**, set on the **Settings** tab via **Edit Configuration**.

The user- and tenant-level actions are configured on the user and tenant profiles (Recording Settings section); the **Auto rules** tab shows the resulting rules read-only.

Per-user announcement settings

By default, user profiles do not show an announcement setting. To expose it for a tenant, enable **Per-user announcement** in the tenant profile's Recording Settings section; the **Recording announcement** setting then appears on that tenant's user profiles.

Custom rules

Custom announcement rules define exceptions — for example, announce external calls only. Click **Add** on the **Custom rules [before]** tab and define the **Tenant**, a **Condition** (match criteria on call attributes or platform-specific metadata), and the **Action** (announce or not).

Avoiding false announcements

By default the recorder evaluates recording rules only after media arrives, so participants can hear an announcement for a call that the rules ultimately decline to record. Enable **Early check of recording filters** (see Recording rules) so the record/no-record decision is made before the announcement is triggered.

Related pages

- Recording rules — the underlying record/no-record decision.
- Tenant configuration — per-tenant announcement mode and the per-user announcement toggle.
- Teams Recording Announcement — platform-specific behavior and setup for Microsoft Teams.

5.4 Recorder servers

Every recorder process in the deployment registers itself with the web portal. The Recording Servers page is the recorder registry plus the recorder settings tree — the place to check recorder health, inspect effective settings, and override settings for an individual recorder.

Where: Administration › Maintenance › Recording Servers.

Recorder registry

The table at the top lists each known recorder with:

- **Recorder name** — click it to open the recorder's own page.
- **Hostname** and **Host IP**.
- **Version** — the recorder software version.
- **Status** — "Service started … ago" / "Service stopped … ago", based on the recorder's reports to the portal.

A recorder's page also shows its call statistics, so you can confirm it is actually processing calls.

> **Note**
>
> The recording license key and its status are tied to the recorder service — see License keys. A stopped recorder also makes the license information unavailable.

Recording settings tree

Below the registry, the **Recording Settings** panel has two tabs:

- **Base Settings** — the settings shared by all recorders, listed as **Category** / **Key** / **Value** rows with **View** and **Override** actions. Categories cover the recorder subsystems: `Audio` (format, AGC, PLC — the same values edited on Audio format), `Filters` (recording rules), `Main`, and one `Protocol::…` category per recording interface (`Protocol::SIP`, `Protocol::SIPREC`, `Protocol::CiscoBiB`, `Protocol::Avaya`, and so on).
- **Overridden Settings** — per-recorder exceptions: **Recorder name** / **Category** / **Key** / **Value** rows with **View** and **Edit**.

Most base settings are maintained through their dedicated admin pages (Recording Interfaces, Recording Rules, Files Location, Audio Files Format); the settings tree shows the resulting key/value store and is the only place to create **per-recorder** overrides.

When to override

Overrides matter in multi-platform deployments where recorders serve different phone systems. For example, to enable early rule evaluation only on a Microsoft Teams recorder, create an override with **Recorder name** = the Teams recorder, **Category** = `Protocol::SIPREC`, **Key** = `EarlyCheckRecordingFilters`, **Value** = `true` — see Recording rules.

Related pages

- Recording interfaces — the base configuration behind the `Protocol::…` categories.
- Audio format — the base configuration behind the `Audio` category.
- License keys — recording license status.
- Troubleshooting — recorder trace logging.

5.5 Live monitoring servers

Live monitoring lets supervisors listen to active conversations in real time. The audio is streamed by dedicated live monitoring servers; this page covers their cluster configuration and health. The customer-facing side — licenses on monitored users and permissions — is documented in Live monitoring in the Administration Guide.

Where: Administration › Maintenance › Live Monitoring Servers.

Cluster settings

The settings shared by all live monitoring servers (edit via **Edit Configuration**):

- **STUN server URL** — one or more STUN servers used for connectivity negotiation.
- **RTP Min Port** / **RTP Max Port** — limits the pool of RTP ports used for audio streams.
- **ICE Lite** — enable to use an ICE Lite agent.

The page also shows the overall live monitoring status with a **Revalidate all** link that re-checks every server.

Server list

Each live monitoring server is listed with:

- **Name**
- **Health status** — **Healthy**, **Unhealthy**, or **Unknown** (the portal periodically probes each server; **Revalidate all** forces a re-check).
- **Service URL** and **Public IP**
- **Version**
- **Sessions** — active, failed, and total session counts; click through a server to inspect individual sessions.

Use **Add** / **Delete** to register or remove servers and **Edit** to change a server's connection details.

Verifying the feature end to end

A healthy server list is necessary but not sufficient — users also need monitoring licenses and permissions. If supervisors report that live monitoring does not work:

1. Check the server **Health status** here (and **Revalidate all**).
2. Confirm the monitored users hold live monitoring licenses — see License usage.
3. Confirm the supervisor's role and group scope allow monitoring — see Live monitoring.

Related pages

- Live monitoring in the Administration Guide — licenses, permissions, prerequisites.
- License usage — monitoring seat assignment per tenant.

5.6 Integrations

The Integrations catalog connects MiaRec to external platforms: telephony integrations bring conversations into the platform through vendor APIs, and single sign-on integrations let users sign in with an external identity provider. This page covers catalog administration and the synchronization jobs at concept level — platform-specific setup lives in the dedicated integration guides.

Where: Administration › System › Integrations. The catalog shows integration cards filtered by **All** / **Single Sign-On** / **Telephony** tabs, with a tenant filter and text search. Each card shows its connection count (or "Not connected") and a **View settings** link; **Add** registers a new connection.

Catalog and connections

An *integration* is a platform entry in the catalog; a *connection* is a configured instance of it — typically one per tenant (a card can hold several connections for different tenants). Opening a card lists its connections, where you can view, edit, and test each one. Some integrations can also be registered as tenant-scoped catalog entries visible to a single tenant.

Which cards appear depends on the build and the enabled feature flags — do not expect an identical catalog on every deployment. Typical entries include Microsoft Teams, Twilio, Webex Native Recording, and a generic "Other phone platform" card on the telephony side, and Google, Microsoft, and Webex on the single sign-on side.

Telephony integrations

Telephony integrations come in two shapes:

- **Recorder-based** — the platform delivers media to the MiaRec recorder (for example, Microsoft Teams via SIPREC). The catalog card handles platform authorization and provisioning; capture itself goes through recording interfaces and recording rules.
- **Sync-based** — MiaRec pulls completed recordings from the platform's own recording service on a schedule (for example, Twilio or Webex Native Recording). Each pull runs as a recurring synchronization job.

Synchronization jobs

Sync-based integrations create jobs such as **Twilio synchronize calls**, **Webex Native Recording synchronize calls**, and **NICE InContact synchronize calls**. They appear in the jobs console like any other job — schedule, runs, processed-record counts, and logs — which is where you monitor and troubleshoot syncing. See Jobs console and the job type reference.

Synced recordings become regular conversations and enter the same processing pipeline as recorded ones. Attribution to tenants and users relies on the integration's user linking and on extension matching — see Tenant configuration.

Single sign-on integrations

SSO cards (Google, Microsoft, Webex) enable the corresponding sign-in button on the login page and account linking for users. SAML identity providers are registered separately under **Administration › User Authentication** — see SAML. The customer-facing view of SSO is covered in Single sign-on in the Administration Guide.

Platform-specific setup

This guide does not repeat per-platform setup steps. Use the dedicated guides:

- Microsoft Teams Integration Guide
- Twilio Flex Integration Guide
- Webex Native Recording Integration Guide
- Five9 Integration Guide

Related pages

- Recording interfaces — recorder-based capture protocols.
- Jobs console — monitoring synchronization jobs.
- Integrations in the Administration Guide — the customer administrator's view of the same catalog.

5.7 Import recordings

The **Import call recordings** job ingests audio files and metadata produced outside the platform — a third-party recorder's archive, a MiaRec export from another system, or a directory of bare audio files. Imported recordings become regular conversations and enter the same processing pipeline as recorded ones.

Where: Administration › Storage › Import Recordings, which opens the jobs console filtered to this job type. Create a job, configure it, and run it manually or on a schedule.

Job settings

- **Data source** — how the job tracks progress across runs: with a continuation token (each run picks up where the previous one stopped) or full mode (re-scan everything).
- **Vendor Format** — the metadata format of the source archive: **MiaRec (*.json)** (the format written by export jobs and replication), **SmartRecord (*.meta)**, **Verba (*.xml)**, **Avaya ACR (*.xml)**, **Crystal Reports (*.xml)**, **RedBox (*.txt)**, **Switchvox (*.xml)**, or bare **MP3 files (*.mp3)** / **WAV files (*.wav)**. For bare audio files, a **REGEX pattern** extracts metadata (dates, phone numbers) from the file names.
- **Source Storage**, **Source Directory**, **Source File Mask** — where to read from; the source is a storage target.
- **Destination Storage**, **Destination Directory**, **Filename format** — where the imported audio files are written and how they are named (the parameterized format described in Files location).
- **Tenant (optional)** — assign all imported recordings to one tenant. Without it, the importer matches tenants from the metadata; enable **Ignore unknown tenants** ("Do not import recordings for unknown tenants") to skip records it cannot match.
- **Metadata only** — import the call records without copying audio files.
- **Remove source files** — delete source files after a successful import (not available together with metadata-only).

The randomization settings (random IDs, phone numbers, dates) exist for generating demo/test data sets from an archive — leave them disabled for production imports.

Run progress, per-record results, and logs are available on the job page like for any job — see Jobs console.

> **Files on network shares**
>
> Source files are read by the MiaRec server processes, not by your browser — a path that opens fine from your workstation can be unreachable from the server. If the archive sits on a network share, either mount the share into the server's local file system (for example, `/mount/backup`) or register it as an SMB storage target, and make sure the service account running the MiaRec background workers has access to it.

Importing encrypted recordings

A MiaRec-format archive exported from an encrypted system contains encrypted audio. Import the matching encryption key before (or along with) the import, or the recordings cannot be played — see Encryption.

Related pages

- Export and backup — producing the archives this job restores.
- Upload settings — end-user manual upload, the other file-based ingestion path.
- Storage targets — defining the source and destination storage.
- Jobs console — runs, records, and logs.

5.8 Upload settings

Manual upload lets end users add audio files through the portal; each uploaded file becomes a regular conversation. As the operator you control whether the feature is available, its limits, where uploaded files are stored, and how metadata is parsed from file names — system-wide defaults plus per-tenant overrides.

Where: Administration › Storage › Upload Recordings. The page shows the system defaults and, in multitenant mode, a per-tenant override table. The customer-facing view is read-only — see Uploading recordings in the Administration Guide.

System defaults

Click **Edit Configuration** to set:

- **Status** — the system-wide policy:
  - **Always enabled for all tenants** / **Always disabled for all tenants** — forced, tenant overrides ignored.
  - **Enabled by default for tenants** / **Disabled by default for tenants** — the default that tenants with status **Default** inherit.
- **Filename format** — the parameterized storage path for uploaded files, using the same parameters as Files location. It must include `%{call-id}`. Default: `%{setup-time#%Y%m%d}/%{setup-time#%Y-%m-%d_%H%M%S}__%{caller-number}__%{callee-number}__%{call-id}`.
- **Maximum file size (in MB)** — default 50.
- **Upload chunk size (in MB)** — default 1; files upload in chunks so interrupted uploads can resume.
- **Storage Target** — the System-level storage target that receives uploaded files.
- **Maximum parallel upload per user** (default 4) and **per tenant** (default 10).

Filename parsing

Uploaded file names can carry metadata (call time, caller/called numbers, an ID). Two parsers fill the conversation fields from the name:

- **Filename REGEX pattern** — a regular expression with named groups such as `year`, `month`, `day`, `hour`, `minute`, `caller_number`, `caller_name`, `called_number`, `called_name`, and `call_id`. A **Test pattern** link lets you try the pattern against sample file names before saving.
- **Parse filename with AI** — an LLM-based parser for file names too irregular for one REGEX. It has the same four-level status values as the upload feature itself (always/default, enabled/disabled), and its own settings: the generative AI **Engine**, the **Instructions** (prompt), and the **Inputs** template.

Per-tenant overrides

In multitenant mode, the table below the system settings lists every tenant with its effective upload status. Open a tenant's row to override:

- **Status** and **Parse filename with AI** — **Enabled** / **Disabled** / **Default** (inherit the system default; forced system statuses always win).
- The tenant's own limits, storage target, and parsing settings — for the AI parser, fields left empty fall back to the system setting values.

Related pages

- Uploading recordings in the Administration Guide — the customer administrator's read-only view.
- Uploading audio files in the User Guide — the end-user procedure.
- Storage targets — destinations for uploaded files.
- Files location — the filename format parameter reference.

5.9 Phone softkey services

Phone softkey services put recording controls on the desk phone itself: through an on-phone menu, a user can trigger on-demand recording (keep or discard the call) and pause/resume recording without opening the web portal. The services are served by the web portal to supported IP phone models — including Cisco, Polycom, Yealink, Aastra, and Panasonic devices.

> **Note**
>
> This feature targets deployments with traditional desk-phone platforms and is rarely used on modern cloud telephony integrations, where recording controls live in the web portal instead.

Where: Administration › System › Phone Softkey Services — a per-tenant status table with **View sessions**, **View**, and **Edit** configuration actions per row. A separate **Phone Softkey Services Sessions** page lists active phone sessions.

Configuration

Per tenant, **Edit** the configuration:

- **Phone softkey services** — **Enable** the feature for the tenant.
- **Max sessions per user** — 1–10; increase it if users have multiple phone devices (default 1).
- **Max session lifetime (days)** — when set, users must sign in from their phones again after the session expires; empty means no expiry.
- **Phones network** — an IP network filter (`x.x.x.x/m`); the services answer only to phones from this network. The default `0.0.0.0/0` allows any address.
- **Authentication** — how users sign in from the phone:
  - **Authenticate users using a PIN**
  - **Authenticate users using a web access password**
  - **Authenticate users using Metaswitch CommPortal**

Sessions

View sessions shows the signed-in phone sessions for a tenant, so you can verify that phones reach the service and are authenticated as the expected users.

Related pages

- Recording rules — the on-demand recording modes the softkeys control.
- Portal authentication — Metaswitch CommPortal authentication.

5.10 Phone number normalization

Phone platforms report numbers in different shapes — `4155551234`, `+14155551234`, `sip:4155551234@pbx` — and inconsistent formats break user matching and make searching painful. Phone number normalization rules rewrite the caller/called numbers reported by the phone system into a consistent format (typically E.164) before conversations are stored, so extension matching and search behave predictably.

Where: Administration › System › Phone Number Normalization. The list shows each rule's **Match Pattern** and **Rewrite To** value, with **Add**, **Delete**, and per-row **Edit**.

Rules

Each rule is a pair:

- **REGEX pattern** — a regular expression matched against the reported phone number.
- **Translation rule** — the replacement, which can reference capture groups from the pattern.

The rules are stored as recorder settings shared by all recorders, and the numbers are rewritten as calls are captured — normalization does not retroactively change existing conversations.

A **Test pattern** page (linked from the rule editor) lets you try a pattern and translation rule against sample numbers before saving. Duplicate match patterns are rejected.

> **Tip**
>
> Normalize toward the same format you use for user extensions (see Extensions) — matching conversations to users compares the reported numbers against the extension list, so both sides must agree on one format.

Related pages

- Extensions — the numbers conversations are matched against.
- Tenant configuration — call-to-tenant association rules.
- Recorder servers — where shared recorder settings live.

6. Storage & data lifecycle

6.1 Files location

The Files Location settings tell the recorder where to write captured audio files and how to name them. The file name format is a parameterized template — with it you control the whole directory layout: group files by date, by tenant-relevant attributes, by IP address, and so on.

Where: Administration › Storage › Files Location. The page shows:

- **Audio files directory** — the root directory for recorded audio, with a free-space gauge for the disk it lives on.
- **Audio file name format** — the parameterized file name template (reference below). Default: `%{setup-time#%Y%m%d}/%{setup-time#%Y%m%d%H%M%S}-%{caller-number}-%{callee-number}-%{call-id}.mp3`
- **File creation mode** — Unix file permissions for the recorded files (Linux only), `000` – `666`, default `664`.

Click **Edit Configuration** to change them. The settings are stored as base recorder settings shared by all recorders; per-recorder overrides are possible under Recorder servers.

File name format reference

The template mixes literal text with parameters:

```
%{parameter-name}
%{parameter-name#format-string}
```

Parameters inside a directory segment create directories (auto-created as needed). If two calls produce the same file name, a unique numeric suffix is appended (`file_2.mp3`, `file_3.mp3`, …).

Example — one sub-directory per day, one file per call:

```
%{setup-time#%Y%m%d}/%{setup-time#%Y-%m-%d-%H%M%S}.mp3
```


Call parameters

Parameter	Description
`%{call-id}`	Unique call ID assigned by MiaRec
`%{parent-call-id}`	ID of the parent call segment (for example, the pre-hold segment on platforms that split held calls)
`%{protocol-call-id}`	Call ID assigned by the phone system (for SIP, the `Call-ID` header)
`%{protocol-tracking-id}`	Interaction/tracking ID assigned by the phone system (Avaya UCID, BroadWorks extTrackingID)
`%{protocol-call-direction}`	Call direction reported by the phone system: `0` unknown, `1` outbound, `2` inbound
`%{call-state}` / `%{record-state}`	Numeric call phase / recording state codes
`%{voip-protocol}`	Numeric code of the capture protocol (SIP, SIPREC, Cisco Built-in-Bridge, …)
`%{setup-time}`	Time the call was established (accepts a `#format` — see time codes below)
`%{alerting-time}` / `%{connect-time}` / `%{disconnect-time}`	Ring / answer / hang-up times (accept `#format`)
`%{duration}` / `%{total-duration}`	Voice duration (connect→disconnect) / total duration (setup→disconnect), seconds
`%{caller-number}` / `%{callee-number}`	Phone numbers of the parties
`%{caller-name}` / `%{callee-name}`	Party names (protocol-dependent; for SIP, from the From/To headers)
`%{caller-id}` / `%{callee-id}`	Party IDs (protocol-dependent; for SIP, the SIP URI)
`%{caller-ip}` / `%{callee-ip}`, `%{caller-port}` / `%{callee-port}`, `%{caller-mac}` / `%{callee-mac}`	IP address, port, and MAC address of the parties
`%{orig-caller-number}` / `%{orig-caller-name}` / `%{orig-callee-number}` / `%{orig-callee-name}`	Original caller / originally dialed party, when different after redirects

Platform-specific parameters

Parameter	Description
`%{sip-header-invite#Header-Name}`	Value of a SIP header from the INVITE, e.g. `%{sip-header-invite#User-Agent}` (initial file creation only)
`%{BroadWorks-userID}` / `%{BroadWorks-groupID}` / `%{BroadWorks-serviceProviderID}`	BroadWorks user / group / service provider IDs
`%{MetaSwitch-recorderParty}` / `%{MetaSwitch-userName}` / `%{MetaSwitch-businessGroup}` / `%{MetaSwitch-systemName}`	Metaswitch CFS attributes
`%{agent-id}` / `%{agent-name}`	Avaya agent ID / name
`%{acd-number}` / `%{acd-name}` / `%{acd-id}`	ACD attributes (BroadWorks and Avaya environments)
`%{transfer-from-*}` / `%{transfer-to-*}` (`number` / `name` / `id`)	Transfer parties (Cisco Skinny protocol only)

Post-processing-only parameters

These are resolved only by post-processing jobs (export, relocate, replication) — they are **not** available to the recorder when the file is first created, because the user/tenant association happens after capture:

Parameter	Description
`%{user-id}` / `%{user-name}`	User the recording is assigned to (first user for internal calls)
`%{group-id}` / `%{group-name}`	The user's group (first group)
`%{tenant-id}` / `%{tenant-name}`	Tenant the recording is assigned to

Conversely, `%{filename}`, `%{filename-full}`, `%{filename-dir}`, and `%{sip-header-invite#…}` are available only at initial file creation, not to post-processing jobs.

Time format codes

Date/time parameters accept a format string after `#`, e.g. `%{setup-time#%Y-%m}` → `2011-02`:

Code	Meaning	Code	Meaning
`%Y` / `%y`	Year with / without century	`%H` / `%I`	Hour, 24-hour / 12-hour
`%m`	Month number (01–12)	`%M`	Minute (00–59)
`%b` / `%B`	Abbreviated / full month name	`%S`	Second (00–59)
`%d`	Day of month (01–31)	`%u`	Microseconds
`%j`	Day of year (001–366)	`%p`	A.M./P.M. indicator
`%a` / `%A`	Abbreviated / full weekday name	`%U` / `%W`	Week of year (Sunday / Monday first)
`%w`	Weekday number (0–6, Sunday = 0)	`%%`	Literal percent sign

File path rewrite rules

Administration › Storage › File Path Rewrite Rules maintains prefix-rewrite rules applied whenever the platform opens a stored file: if a file's stored URL starts with a rule's **Original File Path**, that prefix is replaced with the **Rewrite To** value. Use rewrite rules when recordings were moved outside MiaRec — a changed mount point, a renamed directory, a migrated server — so existing database records still resolve to the files' new location, without touching the database.

To physically move files (and update their records), use the relocate job instead — see File maintenance jobs.

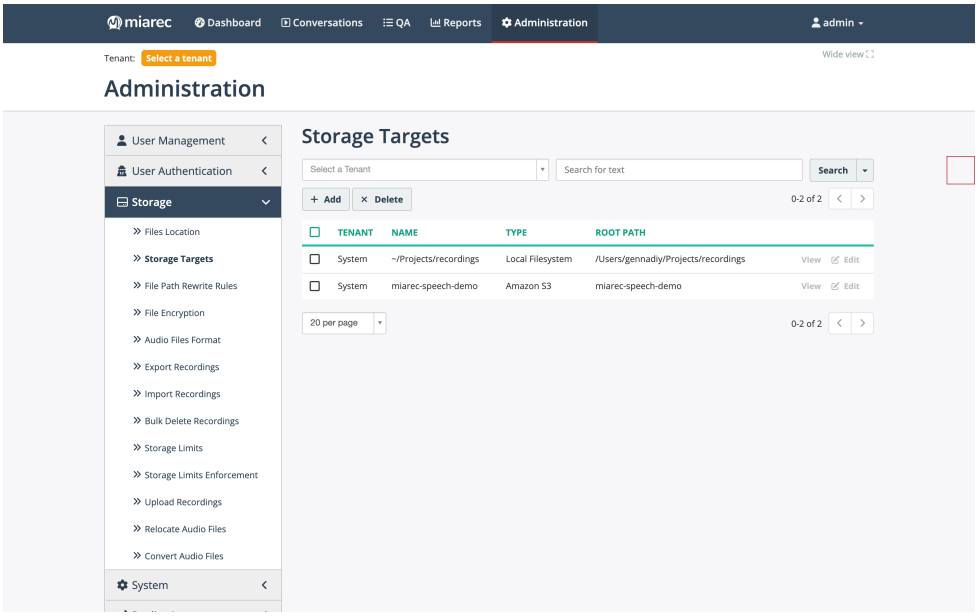
Related pages

- Storage targets — non-local destinations for recordings.
- Audio format — the format of the files being written.
- File maintenance jobs — relocating existing files.

6.2 Storage targets

A storage target is a named connection to a storage destination — a local directory, a network share, an FTP/SFTP server, or an S3 bucket. Targets are referenced everywhere the platform reads or writes files: recording storage, upload settings, export / import jobs, relocation, and replication.

Where: Administration › Storage › Storage Targets. The cross-tenant list shows each target's **Tenant**, **Name**, **Type**, and **Root Path**, with **Add** / **Delete**, a tenant filter, and per-row **View** / **Edit**.



The Storage Targets list. Notice the Tenant column — targets belong to the System tenant or to an individual tenant, and jobs can only use targets visible in their scope.

Storage types

Operators can create targets of all five types:

Type	Settings
Local Filesystem	Base path on the server's file system
Network Share (SMB)	Host, Port, User, Password, Base path, direct TCP option
FTP/FTPS	Host, Port, User, Password, TLS options (require TLS / implicit TLS), Base path
SFTP	Host, Port, User, Password or SSH key, Base path, atomic POSIX rename option
Amazon S3	S3 Bucket, AWS Access Key ID / Secret Access Key, S3 Endpoint URL, Region, AWS Signature Version, server-side encryption option

The S3 type works with any S3-compatible service via the endpoint URL.

> **Customer-visible types**
>
> Tenant administrators can also define storage targets for their own organization, but only of the **FTP/FTPS**, **SFTP**, and **Amazon S3** types — **Local Filesystem** and **Network Share (SMB)** are operator-only, since they expose the server's own file system and network. See Storage targets in the Administration Guide for the customer-facing view.

Tenant scoping

Each target belongs to a tenant. System-tenant targets serve platform-wide functions — for example, the upload feature stores files to a System-level target. Tenant-scoped targets support per-tenant scenarios such as exporting one tenant's recordings to that customer's own S3 bucket.

Adding and testing

Click **Add**, pick the type, fill in the connection settings, and use **Save and Test** to verify the connection immediately. On a saved target's page, **Test a Connection** re-checks reachability and authentication — useful after credential rotations or firewall changes. Secrets are hidden on the view page.

Related pages

- Files location — where the recorder writes new recordings.
- Export and backup / Import recordings — jobs reading from and writing to targets.
- Storage targets in the Administration Guide — what tenant administrators can manage themselves.

6.3 Audio format

The Audio Files Format settings control the codec, channel layout, and audio filters the recorder applies when writing recordings. They trade storage size against audio quality — and they matter for analytics: stereo recordings keep the two parties on separate channels, which improves speaker attribution in transcripts.

Where: Administration › Storage › Audio Files Format; click **Edit Configuration** to change:

- **File format** — **WAV**, **MP3**, or **Auto** (default MP3).
- **Audio format** — **Mono** or **Stereo** (default stereo). Keep stereo where the capture interface delivers the parties separately — channel separation is what lets the transcription engine attribute speech to the right speaker.
- **AGC filter** — Automatic Gain Control normalizes volume levels between the two audio channels (default on), with **AGC maximum gain level** (1.0–5.0, default 3.0) limiting amplification so faint noise is not boosted to full volume.
- **PLC filter** — Packet Loss Concealment improves audio quality under slight packet loss (less than 5%; default on).
- **MP3 bitrate** — 8/16/24/32/64 kbps (default 16). The value applies per audio channel and per 8,000 Hz of sample rate — a stereo file at 16,000 Hz is stored at four times the configured bitrate.
- **MP3 quality** — the encoder's quality/speed trade-off, 0 (best, very slow) to 9 (worst, very fast), default 5.

The settings are stored as base recorder settings shared by all recorders (`Audio` category); use per-recorder overrides under Recorder servers for exceptions.

Changing format for existing recordings

These settings affect newly captured audio only. To re-encode files that already exist — for example, converting old WAV archives to MP3 to reclaim space — run the **Convert audio files** job; see File maintenance jobs.

Related pages

- Files location — where and under what names the files are written.
- Recorder servers — per-recorder audio setting overrides.
- File maintenance jobs — converting existing files.

6.4 Encryption

Audio file encryption stores recordings encrypted at rest, adding a second protection layer on top of role-based access control: even someone with direct access to the storage (or to a backup archive) cannot play the files without the private key. It supports compliance regimes such as PCI-DSS and HIPAA.

Where: Administration › Storage › File Encryption — in multitenant mode, a per-tenant status table plus the encryption key list.

Architecture

Every recording is encrypted with its own random **AES-256** key. That per-file key is in turn encrypted with an **RSA** key pair (2,048-bit): the public key encrypts, and only holders of the private key can decrypt. Private keys are never stored in plain text — even with direct database access, they cannot be retrieved.

In multitenant mode, **each tenant has its own encryption key**. A file leaked across tenants is useless to the other tenant. A tenant can even withhold its private key from the service provider entirely: recordings are then encrypted with the tenant's public key, and nobody on the provider side — including root operators — can play them; playback requires importing the files and the private key into the tenant's own MiaRec instance.

Setup checklist

1. **Create an encryption key** (or import an existing one) — for the System tenant first, then per tenant in multitenant mode. Recordings can only be encrypted once an active key exists.
2. **Export a backup of the key** and store it securely outside MiaRec.
3. **Grant access** to the users who must play encrypted recordings.
4. **Enable audio file encryption** for the tenant (or system-wide in single-tenant mode).

The System tenant needs a key of its own, but does not need encryption *enabled* unless calls are recorded into the System tenant (not recommended).

Managing keys

Click **Add Encryption Key**:

- **Name** — a label for the key.
- **Status** — "Use this encryption key for new recordings" marks it active.
- **Protection mode**:
  - **User credentials** — the private key is protected by the credentials of the users granted access; decrypting requires one of those users. There is no way to obtain the private key without a user's password.
  - **Application credentials** — the private key is protected by the application itself, so the platform can decrypt without an individual user's credentials (needed for unattended decryption, for example export jobs that decrypt).
- **Key length** — or import an existing key pair in PEM format instead of generating one.

The key list shows **Date Created**, **Name**, **Fingerprint**, **Status**, and **Protection mode**.

Export and import

Export (on the key's page) saves a password-protected backup of the key. Export every key and store the backups outside MiaRec — encrypted recordings are unrecoverable if the key is lost (all authorized users forget their passwords, or the database is destroyed).

Import creates a key from an existing exported key pair — used when restoring an archive on a new system, or when a tenant supplies their own key material.

> **Warning**
>
> A lost private key means permanently unplayable recordings. Treat key export backups as part of your disaster-recovery baseline, not an optional extra.

Granting access

With **User credentials** protection, users must be explicitly authorized per key before they can play or download recordings encrypted with it: on the key's page, select users in the **Unauthorized Users** list and click **Grant access** (**Revoke access** reverses it). An administrator — even a root administrator — can only grant access to keys they themselves have access to.

Enabling encryption

In multitenant mode, open the tenant's row in the File Encryption status table and click **Edit Configuration** to enable audio file encryption for that tenant (the same setting is available on the tenant profile). In single-tenant mode, the page has a single configuration. From then on, newly captured recordings are encrypted with the tenant's active key — existing files are not retroactively encrypted.

Playback, downloads, and exports

- **Playback and individual downloads** are decrypted in flight for authorized users; the file leaves the platform unencrypted.
- **Bulk ZIP downloads** are likewise decrypted in flight for users authorized on the relevant keys.
- **Export, relocation, and replication jobs** keep files encrypted by default — safe for backups, since the backup utility needs no decryption ability. The export and relocation jobs have an explicit **Decrypt files** option for when the consumer needs plain audio; see Export and backup.
- **Restoring an encrypted archive** on another system requires importing the matching key — see Import recordings.

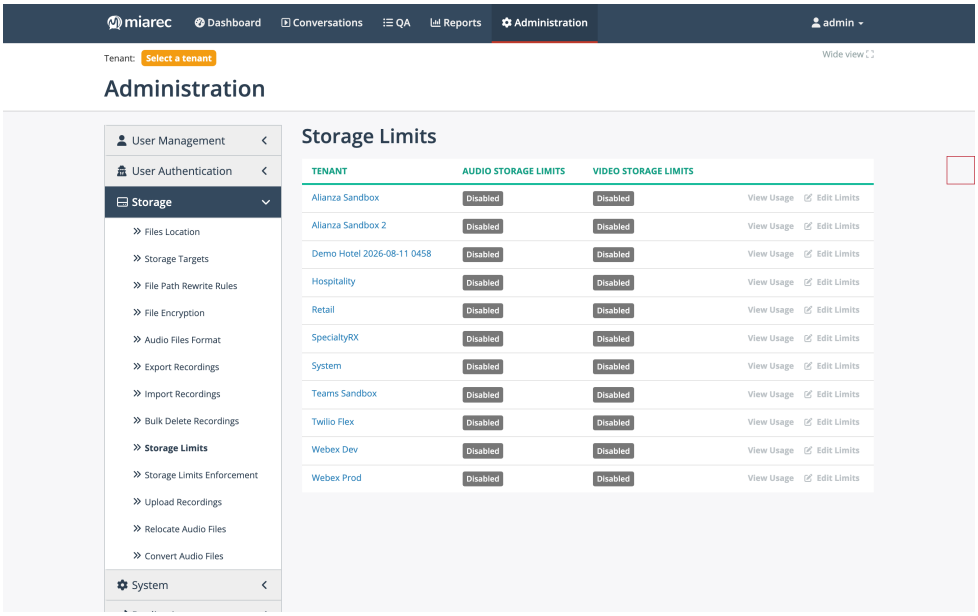
Related pages

- Audio file encryption in the Administration Guide — the tenant administrator's view (their own key and access grants).
- Export and backup — encrypted exports and the Decrypt files option.
- Import recordings — restoring encrypted archives.

6.5 Storage limits

Storage limits cap how much recorded data each tenant can accumulate. When a tenant exceeds a limit, the enforcement job deletes that tenant's oldest recordings until usage is back under the cap — an automatic, rolling retention mechanism per tenant.

Where: Administration › Storage › Storage Limits — a per-tenant table showing the audio and screen-recording limit status (UI columns: **Audio Storage Limits** and **Video Storage Limits**), with **View Usage** and **Edit Limits** per row. The same limits are also editable in the tenant profile — see Tenant configuration.



The per-tenant Storage Limits table. Each row links to that tenant's usage view and limit editor; limits are disabled for all tenants in this environment.

Limit types

Limits are set separately for audio recordings and screen recordings, and each side supports three limit types that can be combined:

- **Size** — total stored bytes.
- **Minutes** — total stored duration.
- **Age (days)** — maximum age of a recording.

View Usage shows the tenant's current consumption against its limits — the same numbers a tenant administrator sees on their Storage usage page.

Enforcement

Limits do nothing by themselves — the **Enforce storage limits** job (**Administration › Storage › Storage Limits Enforcement**) applies them. On each run it processes every tenant with limits enabled, finds recordings exceeding the age limit and the oldest recordings above the size/minutes overage, and deletes their files. Schedule it (for example, nightly) like any other job; runs and logs are in the jobs console.

> **Do-Not-Delete is honored**
>
> The enforcement job always skips conversations flagged **Do-Not-Delete**, regardless of the job's other criteria. This is a hard exclusion built into the job — unlike the bulk delete job, where the exclusion must be part of the job's filters.

Related pages

- Retention and bulk delete — filter-based retention jobs.
- Tenant configuration — the tenant-profile view of the same limits.
- Storage usage in the Administration Guide — what tenant administrators see.

6.6 Retention and bulk delete

Retention policies are implemented with the **Bulk delete recordings** job: a scheduled deletion job whose filters select what to delete — for example, "audio older than 180 days" or "everything from group X after 3 years". You can create several jobs for different policies (one group keeps 3 years, another 7). The same job, run once with narrow filters, is also the tool for one-off bulk deletions.

Where: Administration › Storage › Bulk Delete Recordings, which opens the jobs console filtered to this job type.

Job settings

- **Delete data** — which data types the job removes (combine as needed):
  - **Call metadata** — removes the conversation record entirely, including its transcript, insights, and evaluations.
  - **Audio recording files** — removes only the audio; the conversation record, transcript, and insights remain searchable and reportable. A common policy deletes audio after a year and the metadata much later — the audio is the bulk of the storage, while the analysis keeps its value.
  - **Screen recording metadata** / **Screen recording files** — the same split for screen recordings.
- **Criteria for deleting call recordings** — the filters selecting the affected conversations (age, tenant, group, and other call attributes). **Filters are required** — the job refuses to run without them, so a misconfigured job cannot silently delete everything.
- **Test only** — a test drive: the run logs what would be deleted without modifying data. Use it before enabling a new policy.
- **Schedule** — run manually or on a recurring schedule for continuous retention enforcement.

Run results report the deleted call/file counts and byte totals; per-record details and errors are in the job's logs — see Jobs console.

The Do-Not-Delete flag

Users with the matching permission can flag conversations as **Do-Not-Delete** to protect them (disputes, training sets, compliance).

> **The bulk delete job has no built-in Do-Not-Delete exclusion**
>
> Unlike storage limits enforcement, which always skips flagged conversations, the Bulk delete recordings job deletes whatever its filters select. **You must include the Do-Not-Delete exclusion in every retention job's filtering criteria** — otherwise flagged conversations are deleted like any others. Verify the exclusion with a **Test only** run.
>
> The customer-facing retention page tells tenant administrators to confirm this with their provider — see Retention in the Administration Guide. Make sure your retention jobs honor that expectation.

Choosing the right deletion mechanism

Goal	Mechanism
Cap per-tenant storage automatically	Storage limits + enforcement job
Policy-based retention with filters (age, group, data type)	Bulk delete recordings job (this page)
Move old audio to cheaper storage instead of deleting	Relocate job — File maintenance jobs
Archive before deleting	Export first, then bulk delete

Related pages

- Retention in the Administration Guide — how retention is presented to tenant administrators.
- Storage limits — the automatic per-tenant cap.
- Export and backup — archiving before deletion.
- Jobs console — schedules, runs, and logs.

6.7 Export and backup

The **Export recordings** job copies recordings — audio files and/or metadata — to a storage target. It serves two main purposes: recurring backups (for example, a nightly export to an FTP server or S3 bucket) and data handoff (delivering a tenant's recordings to the customer's own storage).

Where: Administration › Storage › Export Recordings, which opens the jobs console filtered to this job type. Jobs run manually or on a schedule.

Job settings

- **Storage Target** — the destination; see Storage targets.
- **Filename format** — the parameterized path/name template for exported files (see the file name reference). Post-processing parameters such as `%{tenant-name}`, `%{user-name}`, and `%{group-name}` are available here, so exports can be organized by tenant or team.
- **Export data** — **Audio files** and/or **Call metadata**. Metadata is written as MiaRec-format JSON files, which the import job can restore.
- **Skip existing files** — check whether the file already exists at the destination before copying; makes recurring backup runs incremental.
- **Group call segments** — group hold/resume segments of the same call together.
- **Remove after export** — delete records after a successful export, turning the job into move-to-archive; leave off for backups. Not compatible with **Group call segments** (the job rejects the combination).
- **Decrypt files** — decrypt files before exporting them (see below).
- **Filtering Criteria** — which conversations to export (date ranges, tenant, group, and other attributes).
- **Schedule** — for recurring backups.

Runs, processed-record counts, and logs are on the job page — see Jobs console.

Encrypted recordings

By default, encrypted recordings are exported **as-is, still encrypted** — safe for backups, since the destination never holds playable audio and the archive can only be used by importing it together with the encryption key. Enable **Decrypt files** only when the consumer needs plain audio, and treat the destination accordingly. See Encryption.

Restoring

A MiaRec-format export (metadata + audio) is restored with the Import recordings job — point it at the export location, choose the **MiaRec (*.json)** vendor format, and import into the same or a new system. For an encrypted archive, import the encryption key first.

Related pages

- Import recordings — restoring exports.
- Storage targets — export destinations.
- Encryption — encrypted exports and key backups.
- Retention and bulk delete — archiving before deletion.

6.8 File maintenance jobs

Four utility jobs keep the audio file inventory healthy: relocating files to different storage, converting their format, splitting long files by silence, and verifying that every database record still has its file. All are regular jobs — filters, schedules, runs, and logs work as described in Jobs console, and each supports a test-only mode where applicable.

Relocate audio files

Administration › Storage › Relocate Audio Files — the **Relocate audio files** job physically moves recording files to another storage target and updates their database records. Typical uses: tiering old audio to cheaper storage (local disk → S3), or migrating off a retiring server.

Settings:

- **Destination storage target** and **Destination filename format** (the parameterized template; post-processing parameters like `%{tenant-name}` are available).
- **If destination file exists** — how to compare/handle an existing destination file.
- **Remove empty directories** — clean up the source storage as files leave it.
- **Metadata file** — write a JSON metadata file alongside each relocated audio file.
- **Decrypt files** — decrypt during relocation (default keeps encrypted files encrypted; see Encryption).
- **Test only** — log what would be moved without modifying data.

To make *old paths* resolve without moving anything, use file path rewrite rules instead — see Files location.

Convert audio files

Administration › Storage › Convert Audio Files — the **Convert audio files** job re-encodes existing recordings: audio file format (WAV/MP3), channel layout, sampling rate, and MP3 bitrate (8–256 kbps; 0 uses the default). The classic use is converting historical WAV archives to MP3 to reclaim storage. New recordings' format is set separately — see Audio format.

Split audio by silence

The **Split audio by silence** job cuts long recordings into separate conversations at silence boundaries, using voice-activity detection. It is used for ingestion paths that deliver continuous audio containing multiple conversations (for example, imported archives recorded as one file per line). VAD settings control the detection: speech threshold, analysis window size, minimum speech and silence durations, and speech padding. **Push new calls to queue** sends the resulting conversations into the processing pipeline (transcription and analysis); a test-only mode is available.

Check files

Check files verifies that the files referenced by database records actually exist and are readable on their storage. Choose the **Check method** by cost and thoroughness:

- **Check file size** — existence and size only (fastest).
- **Read first 1024 bytes** — confirms the file is readable.
- **Read full file** — full read (slowest, most thorough).

Run it after storage migrations, path rewrites, or suspected storage incidents; failures are reported per record in the job's logs.

Related pages

- Files location — filename templates and path rewrite rules.
- Audio format — the format of newly captured audio.
- Storage targets — relocation destinations.
- Jobs console — runs, records, and logs.

7. Replication

7.1 Replication overview

MiaRec replication copies data from one MiaRec instance to another over HTTPS. It is asynchronous and push-based: jobs on the sending server upload changes to the receiving server's replication API, and the receiving server applies them on arrival. Because every server can be configured both to send and to receive, the topology is multi-master — two sites can replicate to each other, or several sites can push into one central instance.

Typical use cases:

- **Redundant / standby site** — keep a second MiaRec instance continuously up to date so it can take over if the primary site fails.
- **Central archive** — several recording sites push conversations into one central instance where analytics, QA, and reporting run.
- **Migration** — move conversations and configuration from an existing instance to a new one (for example, on-premises to cloud) while the old instance stays in service.

What is replicated

Replication transfers two kinds of data, handled by two different jobs:

- **System configuration** — tenants, users, groups, roles, storage targets, processing queues, jobs, custom fields, report templates, evaluation forms, SAML identity providers, tags, topics, clients, AI tasks, integrations, redaction rules, and speech engines. Related child records (license assignments, group members, permissions, per-tenant settings, and so on) travel with their parent records.
- **Conversation data** — conversation metadata (always), and optionally audio files, evaluation reports, and transcripts.

How it works

Three building blocks cooperate:

- **Replication slots** (sending side) capture configuration changes incrementally. A slot is a named change log: you choose which tables it monitors, and every insert, update, or delete in those tables is recorded until a replication job consumes it. This is what lets the system-configuration job send only what changed instead of rescanning everything.
- **Replication tokens** (receiving side) authenticate and constrain incoming replication. A token defines what the remote server may send (which data types), how conflicts are resolved, and where received audio files are stored.
- **Replication jobs** (sending side) do the actual work: *Replicate System Configuration* pushes configuration records, and *Replicate Recordings* pushes conversations and their files. Both authenticate to the target server with a token ID and secret.

Replication activity is audited on both sides — sent and received records appear in the audit trail.

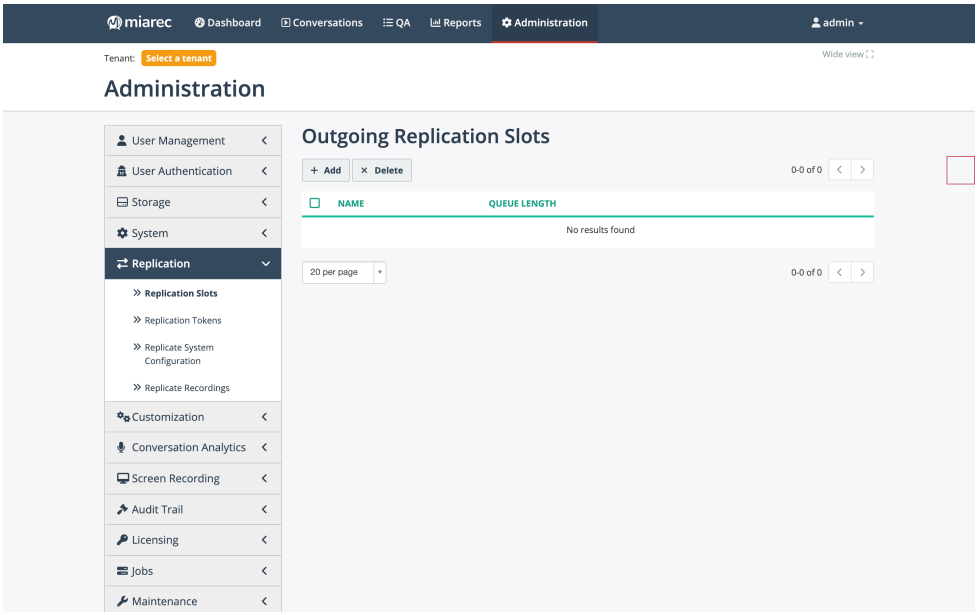
> **Note**
>
> Replication complements, but does not replace, backups. It keeps another live instance up to date; configuration changes — including deletions — propagate to the receiving side, so it is not a point-in-time recovery mechanism.

Where to go next

- Configuration — create the receiving-side token and the sending-side slot.
- Replication jobs — configure and monitor the two jobs, and chain post-processing on the receiving server.

7.2 Configuring replication

Replication is configured in two places: a **replication token** on the receiving server (authorizes and constrains incoming data) and, for incremental system-configuration replication, a **replication slot** on the sending server (captures changes). Both live under **Administration › Replication** and are available in the System tenant only.



The Replication section of the Administration menu on a freshly installed system: Replication Slots (shown) and Replication Tokens start as empty lists until you add them; the two replication jobs have their own menu entries below.

Receiving server: create a replication token

On the server that will *receive* data, go to **Administration › Replication › Replication Tokens** and click **Add**. The token settings are:

- **Status** — *Active*; inactive tokens reject all requests.
- **Name** — a label identifying the sending server.
- **Replication Token** — generate a random secret or enter your own. The sending server presents the token ID and this secret with every request; the secret is stored hashed and cannot be viewed later, so copy it now.
- **Remote IP address** — optionally restrict which source network may use the token.
- **Replicate data** — which data types this token accepts: *System configuration*, *Call metadata*, *Audio files*, *Evaluation Reports*, *Transcripts*. Grant only what the sending server should be allowed to push.
- **Update existing data** — conflict handling when a received record already exists: **Always** (incoming data wins) or **If newer** (the default; the record is updated only if the incoming copy is newer).
- **Related data sync mode** — for a record's related child data: **Merge with existing data** (default) or **Replace with source data**.
- **Call assignment** — whether replicated conversations are matched to local users and clients on arrival.
- **Storage Target**, **Directory**, and **Filename format** — where received audio files are written; see Storage targets.
- **Tenant (optional)** — pin all data received with this token to one tenant account.
- **Publish received records to queue(s)** — push each received conversation into one or more processing queues so the receiving server can run its own pipeline (transcription, AI tasks) on replicated data. See Replication jobs.

Sending server: create a replication slot

A slot is needed when the *Replicate System Configuration* job runs in incremental mode. On the sending server, go to **Administration › Replication › Replication Slots** and click **Add**:

- **Name** — a label for the slot.
- **Monitor changes in data** — the configuration tables to capture: Users, Groups, Roles, Tenants, Processing Queues, Storage Targets, Jobs, Custom Fields, Report Templates, Evaluation Forms, SAML Identity Providers, Tags, Topics, Clients, AI Tasks, Integrations, Redaction Rules, and Speech Engines.

From the moment the slot exists, every insert, update, and delete in the monitored tables is recorded. The slot list shows each slot's **Queue Length** — the number of captured changes not yet consumed by a replication job. A steadily growing queue length means no job is draining the slot.

> **Warning**
>
> Create the slot *before* the first full replication run. Changes made while no slot exists are not captured, so an incremental job that starts later would miss them.

Conversation data does not use slots — the *Replicate Recordings* job tracks its own progress. Continue with Replication jobs.

7.3 Replication jobs

Two jobs on the sending server perform the replication. Both have dedicated entries under **Administration › Replication** (*Replicate System Configuration* and *Replicate Recordings*) and also appear in the jobs console, where runs, processing records, and logs are monitored like any other job.

Both jobs share the connection settings:

- **Target URL** — the receiving server's address (HTTPS), with an option to verify the SSL certificate.
- **Replication token** — the token ID and secret created on the receiving server (see Configuration).

Replicate System Configuration

Pushes configuration records (tenants, users, roles, AI tasks, and the other tables listed in Replication overview) to the target server. The job's **Data source** setting decides how it finds work:

- **Incremental using replication slot** — consume the changes captured by a replication slot. This is the steady-state mode: each run sends only what changed since the previous run.
- Full modes — rescan and send all records. Useful for the initial synchronization or to repair a drifted target.

Run results show **Replicated records** and **Conflicts** (records the receiving side refused to apply, for example because its copy is newer under the *If newer* update strategy).

Replicate Recordings

Pushes conversations to the target server. Call metadata is always sent; the job additionally replicates **audio files**, **evaluation reports**, and **transcripts** according to its *Replicate data* selection (the receiving token must allow the same types). Audio files are uploaded in resumable chunks (5 MB by default), so an interrupted transfer of a large file continues instead of restarting.

Operational behavior worth knowing:

- Conversations still in progress are skipped with a warning (*call is still active*) and picked up on a later run once they complete.
- Already-replicated conversations are skipped, so re-running the job is safe.
- Run results separate **Replicated metadata**, **Replicated files**, file-access warnings, and **Conflicts**.

Like other conversation-processing jobs, the job's **Filtering criteria** limit which conversations are replicated (for example, one tenant only), and its schedule can be continuous or periodic — see the jobs console.

Post-processing chains

Replication plugs into the processing-queue pipeline on both sides:

- On the **sending** server, the *Replicate Recordings* job publishes a *Call - Replication sent* event for each replicated conversation, so a queue can trigger follow-up processing after successful handoff.
- On the **receiving** server, each arriving conversation raises a *Call - Replication received* event, and the replication token's **Publish received records to queue(s)** option feeds received conversations into queues directly. Attach the transcription job to such a queue and the receiving server transcribes and analyzes replicated conversations with its own Conversation Analytics pipeline.

Verifying replication

- Watch the two jobs' **Latest run** / **All runs** tabs for errors and conflicts.
- On the sending server, confirm the replication slot's queue length returns to near zero after each system-configuration run.
- On the receiving server, open the replication token to review the received-data report and any per-record errors recorded for that token.
- Replication send/receive activity is also recorded in the audit trail.

8. Conversation Analytics operations

8.1 Pipeline overview

Conversation Analytics is not one service but a chain of jobs. Each conversation flows through the pipeline stage by stage:

1. **Capture** — a conversation is recorded, imported, or uploaded. When it completes, the platform raises a pipeline event (for example *Call - Recording finished*).
2. **Transcription** — the *Transcribe recordings* job sends the audio to a speech-to-text engine and stores the transcript. See Transcription.
3. **AI tasks** — the *AI Assistant* job runs the enabled AI tasks against the transcript through a generative AI engine and writes the results to their destinations: summary, sentiment scores, topics, custom fields, notes, speaker labels, and — if the tenant uses it — Auto QA evaluations.
4. **Optional stages** — entity recognition tags named entities in the transcript, and data redaction masks sensitive content in audio and transcript.

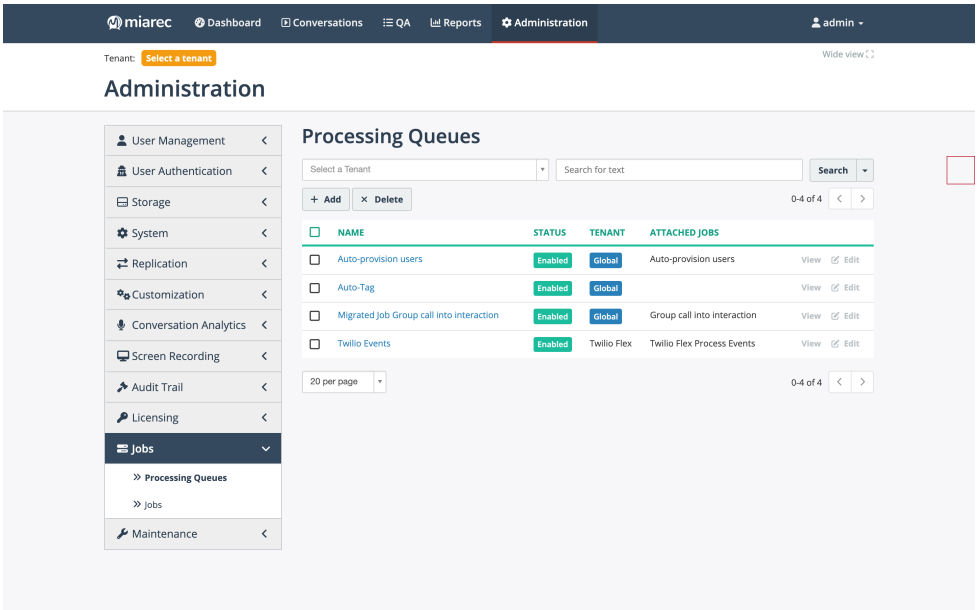
Everything the customer-facing guides describe as "processed automatically" is produced by this chain. The operator's job is to keep it flowing: engines reachable, queues draining, jobs running, limits not exhausted.

Queues connect the stages

Stages are glued together by **processing queues** (**Administration › Jobs › Processing Queues**). A queue is populated by events — *Call - Recording finished*, *Call - Transcription completed*, *Call - Redaction completed*, *Call - Upload completed*, replication events, and so on — and one or more jobs consume it as their data source. A typical chain:

```
recording finished ⟶ [queue] ⟶ Transcribe recordings ⟶ [queue] ⟶ AI Assistant
```

The transcription job's *Action after successful processing* publishes each finished conversation to the next queue, where the AI Assistant job picks it up. The same pattern extends the chain to entity recognition or redaction. Queues, events, and the source/destination/error roles are explained in detail in Pipeline and queues.



The Processing Queues page. Notice the Tenant column (a queue is Global or scoped to one tenant) and the Attached Jobs column showing which job consumes each queue — a queue with no attached job accumulates records that nothing processes.

Jobs do the work

Each stage is a persistent job with its own configuration, filters, schedule, run history, and logs, managed in the jobs console:

Stage	Job type
Transcription	Transcribe recordings
AI tasks (summary, sentiment, topics, custom insights, Auto QA)	AI Assistant
Entity recognition	Recognize entities
Data redaction	Redact data

A job's **Filtering criteria** decide which conversations it processes — this is where licensing is enforced (for example, transcribe only conversations of users holding the Conversation Analytics license). The full grouped catalog, including non-analytics jobs, is in the job type reference.

Where analytics configuration lives

The **Administration › Conversation Analytics** menu bundles configuration, usage metering, job monitoring, and results per feature — each page is tabbed:

Menu item	Tabs	Operator pages
Transcription	Usage / Engines / Jobs / Results	Transcription
AI Tasks	AI Tasks / Global Tasks / Overrides / Usage / Engines / Jobs / Playground	AI tasks, Generative AI engines, Playground
Topic Analysis	topic list	administered per tenant — see the Administration Guide
Entity Recognition	Engines / Jobs	Entity recognition
Data Redaction	Rules / Jobs / Redacted / Highlighted	Data redaction

What can stall the pipeline

- **Engine unreachable or misconfigured** — transcription or AI tasks fail; check the engine's *Test a Connection* and the job logs.
- **Queue with no attached job** — records accumulate and nothing downstream happens.
- **Job filters excluding everything** — the job runs but processes zero records.
- **Tenant limits exhausted** — requests are refused and counted as *denied requests* on the usage tabs; see Tenant configuration.
- **Job stopped or failed** — check Monitoring and troubleshooting for the transcription-focused checklist; the same techniques apply to every pipeline job.

8.2 Transcription

Transcription requirements

Transcription is the first analytics stage: it converts a conversation's audio into the transcript that every downstream AI task depends on. Before configuring it, make sure the licensing, provider, network, and storage prerequisites below are met.

License requirements

Transcription requires a valid license for each user whose conversations are transcribed. The transcription entitlement is part of the **Speech analytics license** (target name: Conversation Analytics license; short code SPCH).

- The license is assigned per user — see Tenant and user configuration.
- The transcription job filters on this license, so conversations of unlicensed users are skipped.
- Per-tenant caps on the number of assignable licenses and on transcription minutes are optional — also covered in Tenant and user configuration.

Speech-to-text providers

Transcription relies on an external speech-to-text service. The supported engine types are:

Engine	Notes
MiaRec Speech API (v1 and v2)	MiaRec's own transcription service. No language selection in MiaRec — language handling is decided by the service. Accepts files up to 250 MB.
Azure Speech-to-Text v3.0	Microsoft Azure. Large per-language picklist plus multi-language auto-detection candidates.
Google Speech-to-Text v1.p1.beta1	Google Cloud. Single-language selection per engine.

Regardless of the provider, the data flow is the same: audio recordings are uploaded to the transcription service, and the transcribed text is returned and stored in the MiaRec database.

Speaker separation depends on the audio: stereo recordings are transcribed per channel (agent and customer separated by design); for mono recordings, the Azure and Google engines offer a "recognize multiple speakers" diarization option.

Network requirements

The MiaRec server must be able to reach the transcription service over the internet (outbound HTTPS). Depending on your environment this may require firewall rules or proxy configuration. Each configured engine has a **Test a Connection** button to verify reachability — see System configuration.

Storage requirements

Transcripts are stored in the MiaRec database. Approximate storage requirement is 3–5 KB per minute of transcribed audio. Factor this into database sizing for high-volume deployments — transcript storage grows with recorded minutes, not with the number of conversations.

Next steps

- System configuration — engine, processing queue, and transcription job.
- Tenant and user configuration — limits and license assignment.

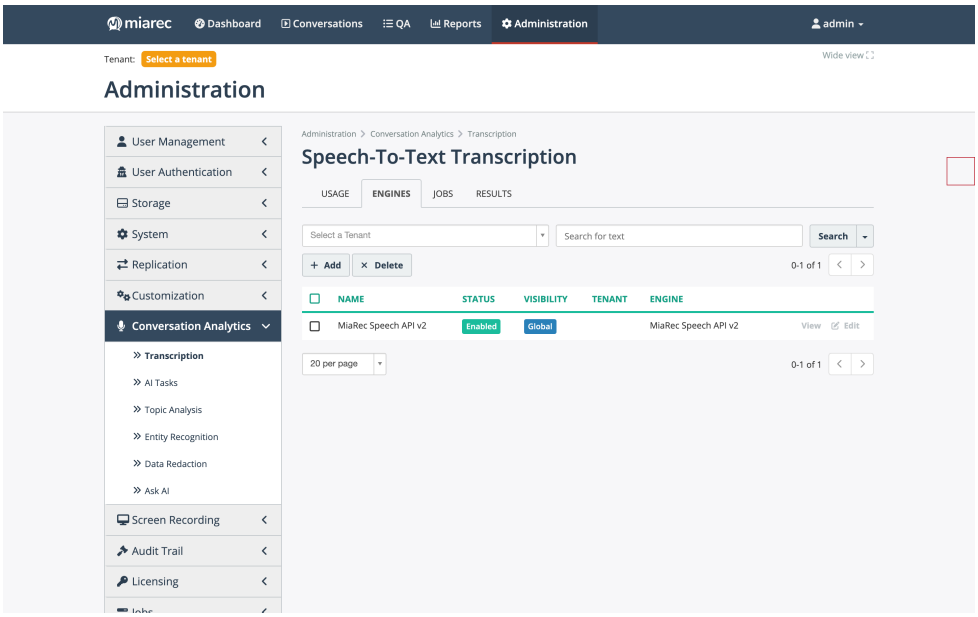
Transcription system configuration

Enabling transcription platform-wide takes three pieces of configuration:

1. A **transcription engine** — the connection to the speech-to-text service.
2. A **processing queue** — collects conversations as they finish recording.
3. A **transcription job** — consumes the queue and sends conversations to the engine.

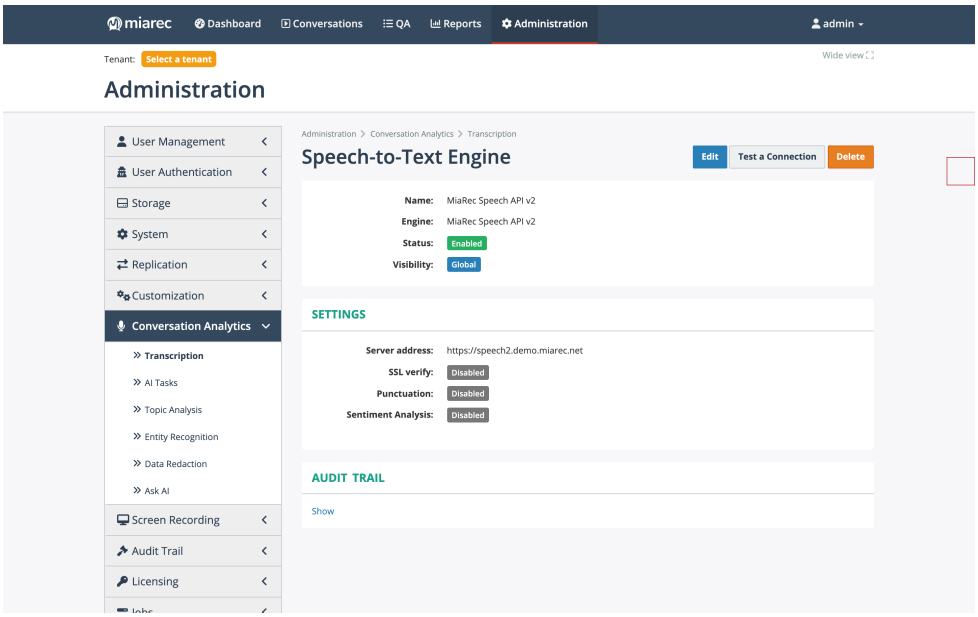
Configure the transcription engine

Go to **Administration › Conversation Analytics › Transcription**, open the **Engines** tab, and click **Add**. Select the engine type (see Requirements for the supported providers) and fill in the provider-specific settings — server address or API keys, region, and feature toggles such as punctuation or sentiment where the engine offers them.



The Engines tab of the Transcription page. Notice the Visibility column: a Global engine serves all tenants; an engine created with a tenant becomes Local to it.

After saving, open the engine and click **Test a Connection**. A sample audio file is sent to the transcription service and the transcribed text is displayed — this confirms credentials, network path, and service health in one step.



A MiaRec Speech API v2 engine. Notice the Test a Connection button and the inline audit trail — every change to engine settings is recorded.

Configure the processing queue

Go to **Administration › Jobs › Processing Queues** and click **Add** to create a queue for transcription. Select:

- **Scope**: Global
- **Record type**: Call
- **Populate queue with events**: *Call - Recording finished*

With this configuration, conversations are added to the queue as soon as recording finishes.

> **Note**
>
> With the global scope, records from all tenants enter the queue, including tenants without transcription enabled. The transcription job (configured next) filters and processes only records belonging to tenants and users with transcription enabled.

Configure the transcription job

Open the **Jobs** tab of the Transcription page and click **Add** to create a *Transcribe recordings* job. Select:

- **Access scope**: Unrestricted - All tenants, including System
- **Data source**: Incremental using queue
- **Source queue**: the processing queue created above
- **Transcription engine**: the engine created above

Under **Filtering Criteria**, add filters so that only licensed conversations are processed:

- **User - License** is **Speech analytics license**
- **Call - Duration** between **0:15** and **2:00:00** (15 seconds to 2 hours)

> **Tip**
>
> The duration filter is optional but recommended: very short conversations rarely produce useful transcripts, and very long ones consume minutes disproportionately.

Under **Action after successful processing**, optionally chain the next pipeline stage:

- **Publish to queue(s)**: the queue consumed by the *AI Assistant* job (summaries, sentiment, topics, Auto QA — see Pipeline overview).

Finally, under **Schedule**, select **Run this job: Continuously**. The job then processes new conversations automatically as they enter the queue.

Per-tenant transcription hint

A tenant-level **Transcription Prompt** (up to 128 characters) can be fed to the engine as a hint — for example, the company name spoken in the greeting, so it is transcribed correctly. It is edited in the tenant's settings and shown as a column in the tenant list. See Tenant and user configuration.

Verify

After the first conversations complete, check the **Results** tab of the Transcription page — it lists all transcriptions with status, conversation date, duration, and parties. Then continue to Monitoring and troubleshooting.

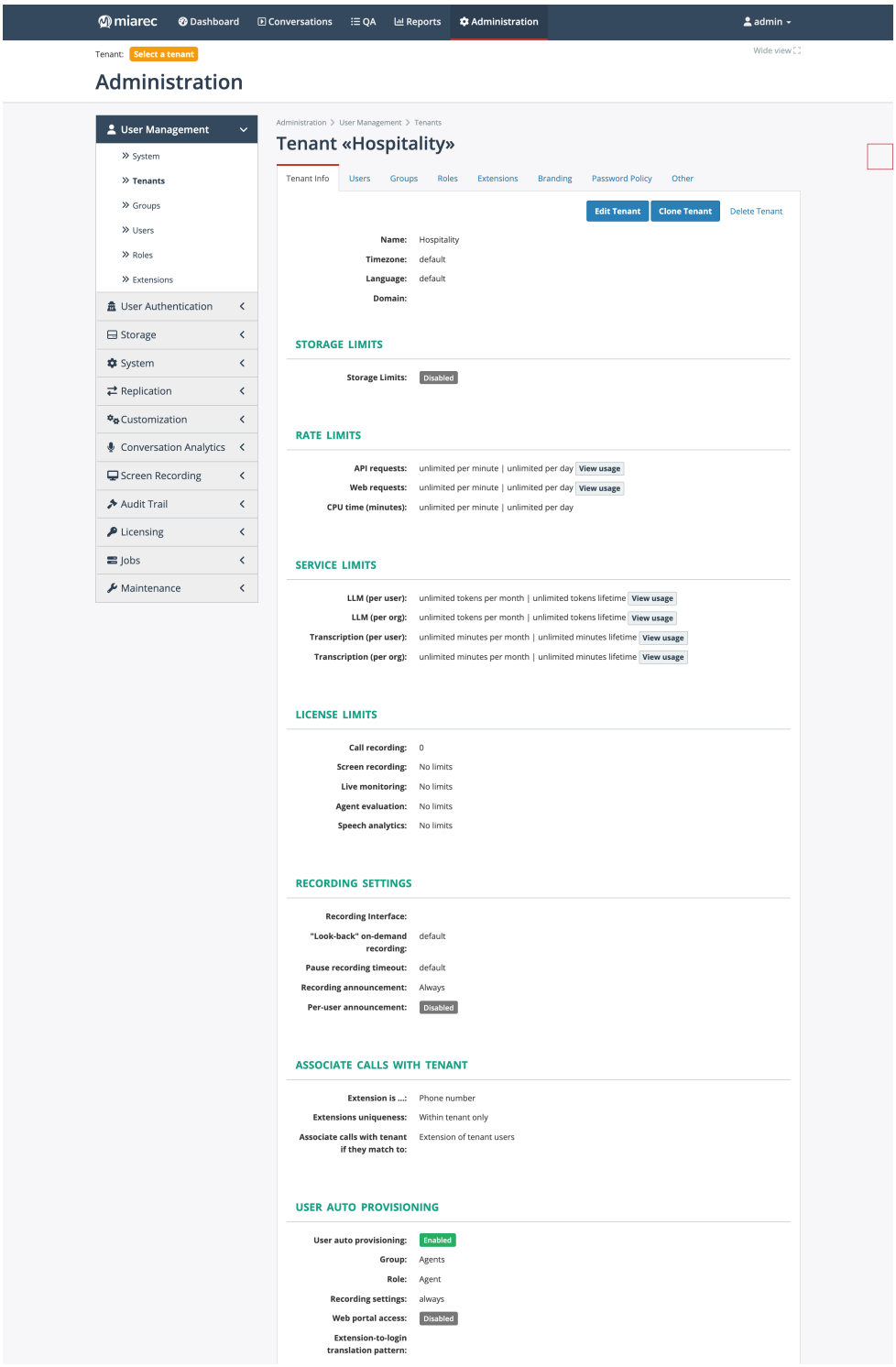
Tenant and user configuration

With the system-wide pieces in place (System configuration), transcription is switched on per tenant and per user:

- Tenant **service limits** (transcription minutes) — optional
- Tenant **license limits** (number of licenses) — optional
- **User license** — required

Tenant configuration

Under **Administration › User Management › Tenants**, open the tenant. The **Tenant Info** tab carries both kinds of limits.



A tenant page. Notice the Service Limits section (LLM tokens and transcription minutes, each per user and per organization, monthly and lifetime, with View usage links) and the License Limits section below it.

Both service and license limits are optional. If not set, the tenant has unlimited transcription usage, provided its users hold valid licenses. Setting reasonable service limits is recommended to avoid

unexpected costs: for reference, an average business user consumes approximately 1,000–1,500 transcription minutes per month; a BPO contact-center agent may use up to 5,000–6,000 minutes per month.

SERVICE LIMITS

Four independent knobs cap transcription minutes:

- Monthly limits (reset at the beginning of each month):
  - **Transcription (minutes/user per month)** — maximum minutes per licensed user each month.
  - **Transcription (minutes/org per month)** — maximum minutes for the whole tenant each month.
- Lifetime limits (do not reset):
  - **Transcription (minutes/user lifetime)** and **Transcription (minutes/org lifetime)**.

Any combination can be set; whichever limit is reached first blocks further transcription requests. Refused requests are counted as *denied requests* on the usage dashboard (see Monitoring and troubleshooting) until the monthly limit resets or the lifetime limit is raised.

> **Note**
>
> Lifetime limits exist for convenience but are not commonly used in practice. Monthly limits align better with billing cycles.

> **Pooled user limits**
>
> User-level limits are pooled across all licensed users in the tenant. If a tenant has 10 licensed users with a limit of 2,000 minutes/user per month, the tenant can use 20,000 minutes per month in total — regardless of how the usage is distributed between users. When an org-level limit is also set, it acts as an additional cap: with an org limit of 15,000 minutes, the tenant gets 15,000 minutes even though the pooled user limit is 20,000.

The same four-knob pattern (per user / per org, monthly / lifetime) also governs **LLM tokens** for AI tasks.

LICENSE LIMITS

The **License limits** section caps how many licenses of each type can be assigned in the tenant; the row for transcription is **Speech analytics** (target name: Conversation Analytics license). Without a tenant-level cap, any number of users can hold the license.

If more users have the license assigned than the tenant's cap allows, the system accepts only the first users up to the cap (in an internally determined order) and declines transcription for the rest — the License usage page shows assigned-versus-limit counts per tenant.

TRANSCRIPTION PROMPT

The tenant's **Transcription Prompt** (up to 128 characters) is an optional hint passed to the speech-to-text engine — for example, a company name spoken in the call greeting, like "This is Acme Company", so uncommon names are transcribed correctly. It is edited in the tenant's transcription settings and appears as a column in the tenant list.

User configuration

Under **Administration › User Management › Users**, open the user and assign the **Speech analytics license**. Only conversations of licensed users are transcribed — the license filter on the transcription job enforces this.

Access control

For users to *see* transcripts, their role needs the appropriate permissions (**Call attributes - Transcript**: View, plus view/playback on the conversation resources). Role configuration is a tenant-administration topic — see Roles and permissions in the Administration Guide.

Verification checklist

- In the tenant profile, are transcription **Service Limits** set (optional but recommended)?
- In the tenant profile, are **License limits** set for Speech analytics (optional)?
- Is the **Speech analytics license** assigned to each user whose conversations must be transcribed?
- Do the users' roles grant access to the **Call attributes - Transcript** resource?

Transcription monitoring and troubleshooting

The Transcription page (**Administration › Conversation Analytics › Transcription**) bundles everything needed to keep transcription healthy: the **Usage** dashboard, the **Jobs** tab for run history and logs, and the **Results** tab listing individual transcriptions.

Monitoring usage

Open the **Usage** tab. As the system operator you can view usage across all tenants or select one tenant; tenant administrators see the same dashboard scoped to their own organization. The dashboard shows:

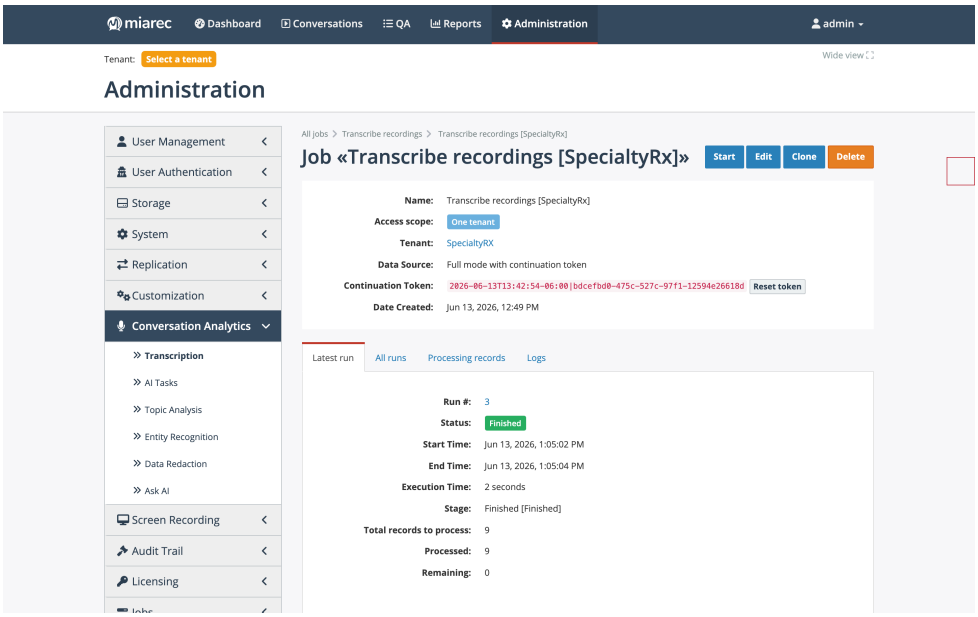
- **Used minutes** and **denied requests** counters for the selected period.
- Where limits are configured: licensed-user counts, the per-user and total monthly limits, current consumption against them, and days remaining until the monthly reset.
- A historical chart of daily minutes and denied requests.

A non-zero *denied requests* count means the tenant hit a limit and conversations are going untranscribed — see Tenant and user configuration for the limit knobs.

> **Note**
>
> Usage is metered when the engine processes a request. Conversations skipped by job filters (no license, duration outside the filter range) consume nothing and appear nowhere on this dashboard.

Monitoring the transcription job

Open the **Jobs** tab and click the transcription job.



A "Transcribe recordings" job. Notice the Latest run tab: status, execution time, and the records total/processed/remaining counters, plus the four sub-tabs (Latest run, All runs, Processing records, Logs) used throughout this page.

Latest run shows the most recent run: how many records were uploaded for transcription, completed, or still pending.

> **Note**
>
> Transcription is asynchronous. Audio is uploaded to the transcription service and the text comes back later, so there is normally a delay between a conversation entering the queue and its transcript becoming available. For uploaded records the job checks periodically for results, up to 50 times; a record whose result never arrives in that window is marked **Failed**.

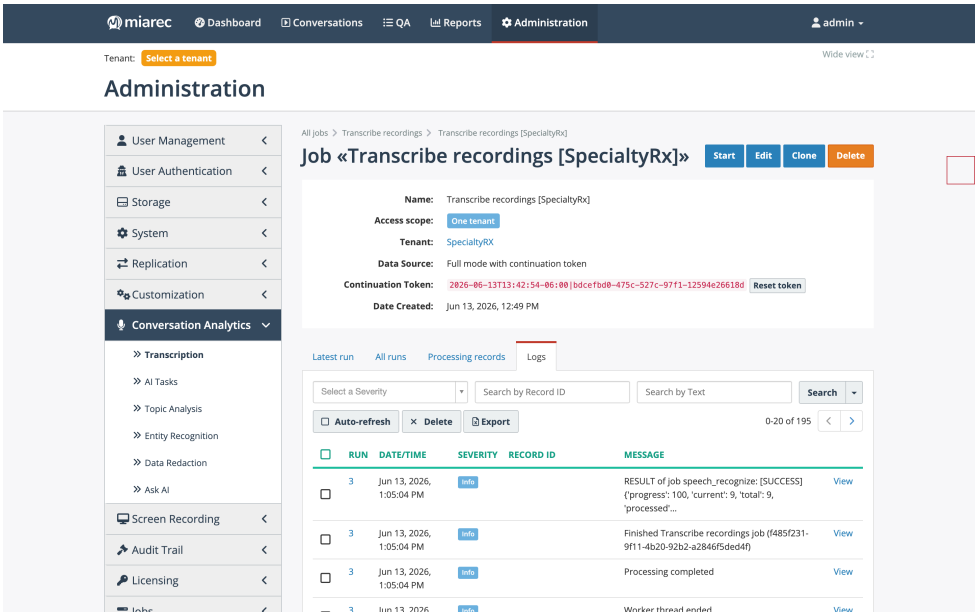
All runs shows the run history with a processed-per-day chart and three metrics:

- **Processed** — records successfully transcribed.
- **Errors** — records that failed (invalid license, network issues, and so on).
- **Recovered errors** — records that initially failed but succeeded on retry.

> **Info**
>
> Because the job polls for asynchronous results, each "not ready yet" check counts as a retry — a high **Recovered errors** number is normal behavior, not a problem.

Processing records lists individual records with totals for Completed, In Progress, Failed, Aborted, and Skipped, filterable by date, status, and record ID. The **Record ID** is the conversation's unique identifier — click it to open the conversation and view the transcript, or click **View** on a row to see the processing details including any error text.

Logs shows recent log messages for the job, with severity filtering, per-record search, auto-refresh, and export.



The Logs tab of a transcription job. Notice the severity filter and the Search by Record ID field — the fastest way to find out what happened to one specific conversation.

Checking results

The **Results** tab lists all transcriptions with created time, status, conversation date, duration, and parties — a quick way to confirm that transcripts are being produced across tenants without opening the job.

Troubleshooting checklist

- Did the job run recently, and with what status? Check the **Latest run** tab.
- Any errors or warnings in the job's **Logs** tab?
- If the job is not consuming work: how many records are pending in the processing queue? **Administration › Jobs › Processing Queues** shows each queue's backlog.
- If the job runs but transcribes nothing: are its **Filtering Criteria** excluding all records?
- Are users assigned the **Speech analytics license**?
- Has the tenant hit its transcription limits? Check the **Usage** tab for denied requests and the tenant profile for the limits.
- Does the engine still reach the service? Open the engine and click **Test a Connection** (see System configuration).

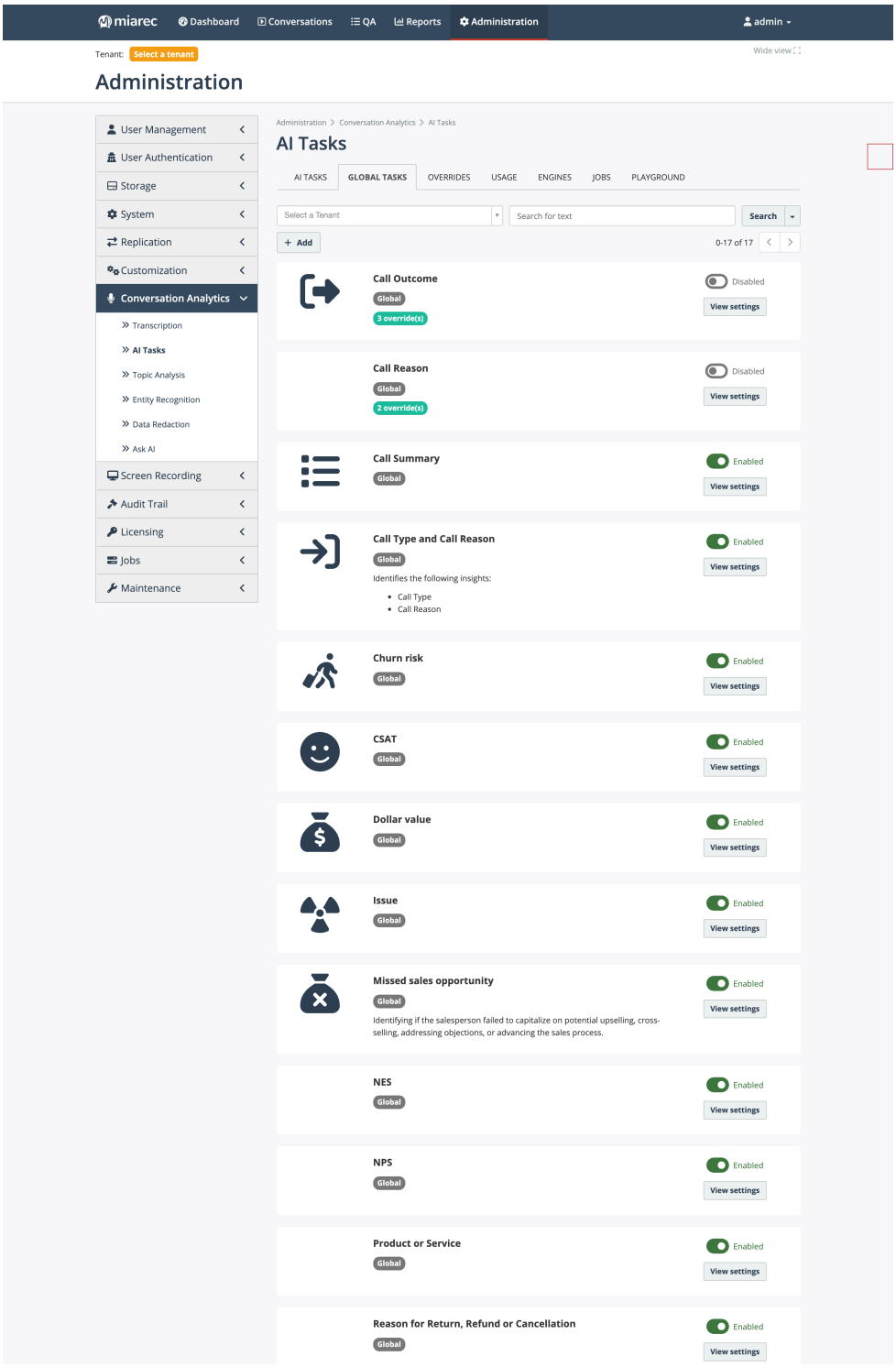
8.3 AI tasks (operator view)

An **AI task** couples a prompt with a *destination* that routes the AI's output into product data — a summary, sentiment scores, topics, custom fields, a note, speaker labels, or an Auto QA evaluation. What tenant administrators can do with tasks is covered in the Administration Guide; this page covers the operator side: global tasks, per-tenant overrides, execution, and token limits.

Everything lives on one tabbed page: **Administration › Conversation Analytics › AI Tasks**.

Global tasks

The **Global Tasks** tab holds tasks defined in the System tenant with global visibility — the shared catalog every tenant receives (conversation summary, sentiment score, topic analysis, CX-metric extractors, and so on). Each card shows the task's enabled state and how many tenants have overridden it.



The Global Tasks tab. Notice the per-task Enabled/Disabled toggles and the override count badges (for example "3 override(s)" on Call Outcome) — the quickest way to see which global defaults tenants have customized.

Maintaining the global catalog is an operator responsibility: a well-curated set of global tasks means new tenants get working analytics with zero per-tenant configuration.

Per-tenant state and overrides

- The **AI Tasks** tab is a per-tenant rollup: for each tenant, how many global tasks apply, how many are overridden, and how many tenant-local tasks exist.
- The **Overrides** tab lists every override: tenant, task, status, whether the prompt is Default/Overridden, and filters. An override toggles enablement, the prompt, and the filtering criteria independently of the global definition — the task's type, destination, and engine remain fixed.

Tenant administrators create their own overrides and local tasks from the same page in their scope; use the Overrides tab to review what tenants changed before you modify a global prompt they may depend on.

How tasks execute

Tasks are executed by the **AI Assistant** job (see the pipeline overview), which typically consumes the queue fed by the transcription job. Per conversation, the job runs every applicable enabled task, in a fixed destination order — speaker labels are resolved first (so later prompts see labeled speakers) and Auto QA runs last. The job itself carries:

- **AI Assist engine** — the default generative AI engine; an individual task can pin a different engine.
- **Process tasks** — all tasks, or an explicit selection.
- **Filtering criteria** — which conversations are processed. Only conversations of users holding the Speech analytics license are scored; unlicensed conversations are counted as license errors.

Task prompts split into *Task instructions* and *Task inputs* (which must contain `${transcript}`), each capped at 8 KB, with a configurable response token limit (default 8,192, maximum 64,000). Long silences in the conversation are marked in the transcript sent to the AI.

Usage and token limits

The **Usage** tab meters LLM consumption per tenant: **used tokens**, **tokens/minute**, and **denied requests**, with a historical chart. Token accounting weights input and output tokens by the engine's cost ratios (see Generative AI engines).

Limits are set on the tenant page (**Service limits**) with the same four knobs as transcription minutes: LLM tokens per user / per organization, monthly / lifetime — pooled semantics included (see Tenant configuration). When a limit is exhausted, the request is refused *before* the engine is called and counted as a denied request; the affected conversations are simply not analyzed until the limit resets or is raised.

Monitoring

- The **Jobs** tab shows the AI Assistant job(s) with runs, processing records, and logs — the same monitoring workflow as transcription.
- The **Playground** tab opens the AI Playground for testing prompts against real conversations without persisting anything.

Related pages

- Generative AI engines — the engines tasks run on (operator-only).
- Auto QA operations — the Auto QA destination end to end.
- AI tasks in the Administration Guide — the tenant administrator's view of the same page.

8.4 Generative AI engines

A **generative AI engine** is a configured connection to a large language model service. Every AI task, Auto QA evaluation, and topic extraction runs through one of these engines. Engine management is an operator-only responsibility: customers do not bring their own LLM — the operator provisions the engines, and tenants simply consume them.

Engines are managed on the **Engines** tab of **Administration › Conversation Analytics › AI Tasks**.

Engine types

Two vendor types are supported:

Type	Default model	Notes
OpenAI-compatible API	`gpt-4o-mini`	The **API Server URL** is configurable, so any provider exposing an OpenAI-compatible endpoint can be used — OpenAI itself, a cloud gateway, or a self-hosted model server.
Google Vertex AI	`gemini-2.5-flash`	Authenticates with a service-account key file or an access token; project ID and location are configurable.

There is no other vendor integration — anything reachable through an OpenAI-compatible endpoint is covered by the first type.

Common settings per engine: name, status (Enabled/Disabled), model name, sampling temperature, presence and frequency penalties, and vendor-specific options (reasoning effort for OpenAI-compatible engines, thinking budget for Vertex AI). API keys and other secrets are stored encrypted. After saving, use **Test a Connection** to verify credentials and reachability.

Token cost ratios

Each engine carries an **input token cost ratio** (default 1.0) and an **output token cost ratio** (default 4.0). Metered usage is calculated as `input tokens × input ratio + output tokens × output ratio`, so tenants' token limits and the usage dashboards account for output tokens being more expensive than input. Adjust the ratios to match the pricing of the model behind the engine.

Visibility and curation

The **Visibility** column shows each engine's scope:

- **Global** — the engine is available to all tenants; it appears in every tenant's task configuration wherever an engine can be selected.
- **Local** — the engine is bound to one tenant (the *Tenant* column names it).

Because global engines are visible across the platform, keep the list curated: use clear, meaningful names and disable or delete experiments. Per-tenant engine assignment — creating a Local engine for one tenant — is supported but uncommon; the typical deployment runs all tenants on a small set of global engines.

Selecting which engine runs the work

- The **AI Assistant job** carries the default engine (its *AI Assist engine* setting) used for every task it executes.
- An individual AI task can pin its own engine, overriding the job default for that task only.

When replacing an engine (for example, moving to a new model), add the new engine, repoint the job (and any task-level pins), verify results in the Playground, and only then disable the old engine — a disabled or deleted engine that a job still references makes the job fail.

Related pages

- AI tasks (operator view) — the tasks that run on these engines, and token limits.
- Playground — test a prompt against a specific engine and conversation.
- Pipeline overview — where the AI Assistant job sits in the chain.

8.5 AI Playground

The **AI Playground** runs a prompt against a real conversation's transcript, live, without persisting anything to the conversation. It is the prompt-design surface for the whole analytics stack: use it to validate a new global task, debug a tenant's override, or compare engines before a switchover.

The playground opens from the **Playground** tab of **Administration › Conversation Analytics › AI Tasks**, and from the **Save and Test** / **Test** buttons on AI task pages.

Experiment types

Five playground experiments exist, one per prompt style:

Experiment	What it tests
Generic AI task	A free-form prompt with a text/JSON response — the general-purpose playground.
Auto QA form	An Auto QA prompt scored against a selected scorecard's questions.
Custom field	A prompt that extracts one or more custom-field values.
Sentiment score	The sentiment-scoring prompt.
Topic form	LLM topic extraction against your topic definitions.

Workflow

1. **Pick a conversation** that has a transcript — the playground's first step is a filtered conversation search.
2. **Edit the prompt** — the task instructions and inputs, response settings, and the engine to run on.
3. **Run** — the prompt executes against the conversation live and the result is displayed.
4. Iterate, then optionally **Save AI Prompt** to write the tuned prompt back to the AI task (or the tenant's override) it belongs to.

Nothing is written to the conversation itself: no summary, score, topic, or evaluation is stored by a playground run. The only persisted change is the prompt, and only when you explicitly save it.

Permissions

Playground access requires permission to view transcripts, plus the permission for the experiment's resource (for example, AI task or evaluation-form access for those experiment types). Tenant administrators have the same playground within their own scope — see AI Playground in the Administration Guide.

Operator use cases

- **Validating a global prompt change** — before editing a global task that tenants rely on, reproduce the current behavior in the playground, apply the change, and compare outputs across a few representative conversations from different tenants.
- **Debugging a tenant complaint** ("the summary misses X") — open the playground, load one of the tenant's conversations, and iterate on the override prompt until the output is right, then save it to the override.
- **Engine comparison** — run the same prompt and conversation against two engines before repointing the AI Assistant job.

8.6 Entity recognition

Entity recognition (NER) detects named entities — people, organizations, dates, money amounts, and the like — in transcripts. Detected entities are stored on the transcript's words with a confidence value, can be queried in the transcript search grammar (`#ENTITY` tokens), and feed data redaction rules that target entity types such as card numbers.

NER is administered under **Administration › Conversation Analytics › Entity Recognition**, a page with two tabs: **Engines** and **Jobs**.

Engines

Two engine types are supported:

Engine	Languages
MiaRec NER API v1	en-US
Amazon Detect PII	en, es

Engines follow the same model as the other analytics engines: name, status, Global or tenant-scoped visibility, and encrypted credentials — see the pattern described in Generative AI engines.

The entity types

The built-in vocabulary is 18 entity types: CARDINAL, DATE, EVENT, FAC, GPE, LANGUAGE, LAW, LOC, MONEY, NORP, ORDINAL, ORG, PERCENT, PERSON, PRODUCT, QUANTITY, TIME, WORK_OF_ART.

The NER job

The **Jobs** tab manages the *Recognize entities* job. Its settings follow the standard pipeline-job shape (source data, filtering criteria, post-processing, schedule — see the pipeline overview), plus NER-specific options:

- **Entity Recognition engine** — which configured engine to use.
- **Recognize entities** — all entity types, or a selected subset.
- **Test only** — run detection without storing results.

Chain the job after transcription (its input must have transcripts) and, when redaction depends on entities, before the *Redact data* job.

> **Note**
>
> As an alternative to the dedicated job, entity recognition can also run as an AI task with the entity-recognition destination, using a generative AI engine instead of a dedicated NER engine.

The user-facing panel flag

Detected entities are shown to end users in a "Recognized Named Entities" panel on the conversation page, with click-to-seek playback. This panel is controlled by the **Named Entity Recognition panel** flag in Advanced Settings. The administration pages above stay visible regardless of the flag — with the flag off, entities are still detected and usable by redaction and search, but users do not see the panel.

8.7 Data redaction operations

The *Redact data* job removes sensitive content from conversations that match redaction rules: matched audio intervals are replaced with silence and matched transcript words are replaced with a mask (by default `*********`). Audio redaction is destructive — the file is re-encoded (and re-encrypted where encryption is on) without the redacted content.

This page covers running and monitoring redaction. **Rule authoring** — pattern syntax, testing, and the review workflow — is covered by the dedicated Data Redaction User Guide, and the tenant administrator's view is in the Administration Guide.

Everything lives under **Administration › Conversation Analytics › Data Redaction**, with four tabs: **Rules**, **Jobs**, **Redacted**, and **Highlighted**.

The redaction job

The **Jobs** tab manages *Redact data* jobs. Besides the standard pipeline-job sections (data source, filtering criteria, post-processing, schedule), the redaction-specific settings are:

- **Action** — one of three modes:
  - **Redact data based on rules** — match and redact in one pass (fully automatic).
  - **Highlight moments based on rules** — detect and record matches without modifying anything (non-destructive dry run / review mode).
  - **Redact the previously highlighted moments** — apply redaction to moments recorded by an earlier highlight run.
- **Rules** — apply all **Active rules**, or an explicit list of **Selected rules**.
- **Limits** — safety caps against runaway rules: **Maximum redacted words per moment** and **Maximum redacted moments per call** (both default 50).

The highlight-then-redact pair supports a human-review workflow: run a highlight job, review the detected moments on the **Highlighted** tab (entries there can be deleted to exclude false positives), then run a job with *Redact the previously highlighted moments*.

Because redaction alters audio and transcripts irreversibly, prefer this staged approach when introducing a new rule; a rule can also be tested against a single conversation without modifying files (see the Data Redaction User Guide).

History: Redacted and Highlighted

- **Redacted** — the log of applied redactions: which conversations were modified and the moments removed.
- **Highlighted** — moments detected by highlight runs that are pending review or redaction; rows can be deleted to prevent their redaction.

Pipeline placement

Redaction operates on transcripts (and audio), so the job runs after transcription — typically consuming a queue fed by the transcription job. When rules reference entity types, run entity recognition first. A completed redaction raises the *Call - Redaction completed* pipeline event, so downstream stages can be chained to run only on redacted conversations — see the pipeline overview.

> **Note**
>
> Rules are defined per tenant or at the System level (the Rules tab shows the tenant of each rule). The job honors its own scope and filters, so one global job can serve all tenants' active rules, or separate jobs can isolate tenants with special requirements.

8.8 MCP server

The MCP (Model Context Protocol) server exposes MiaRec data to external AI clients — desktop AI assistants, agent frameworks, or custom integrations that speak MCP. A connected client can query conversation reports, transcripts, and QA metadata through a fixed set of read-only tools; there are no write tools, so an AI client can never modify MiaRec data.

Client access

- **Endpoint**: `POST /api/mcp` on the web portal (JSON-RPC).
- **Authentication**: HTTP Basic with a MiaRec user account. The account's role must have the **REST API** permission — the same permission that gates the REST API (see Sessions and API usage for monitoring).
- **Scoping**: every tool call runs as the authenticated user. The client sees exactly the conversations, users, and groups that user's role and access scope allow — nothing more. To scope an AI client to one tenant or one group, create a dedicated user with a suitably restricted role.

API rate limits and per-tenant API request quotas apply to MCP traffic like any other API traffic.

Available tools

The server exposes 12 read-only tools:

Tool	Purpose
`build_report`	Build a report — limited to the *Calls Summary* and *Call Details* report types.
`list_report_types`	Discover the report types available to `build_report`.
`list_report_columns`	Discover the columns available for a report type.
`list_filters` / `list_filter_operators` / `list_filter_choices`	Discover the filters, operators, and value choices usable in report queries.
`get_conversation_transcript`	Fetch one conversation's transcript.
`list_topics`	List the configured topics.
`list_users` / `list_groups`	List users and groups (within the caller's scope).
`list_evaluation_forms` / `get_evaluation_form`	List and fetch scorecards (evaluation forms).

Restricting exposed columns and filters

Administration › System › MCP Server controls how much of the reporting surface MCP clients can see. For each supported report type (*Calls Summary* and *Call Details*) the page shows the report's columns in category groups — including the AI-produced custom-field columns — with checkboxes, plus a list of visible filters.

Only the columns and filters selected here are discoverable and usable through the MCP tools. Use this to keep sensitive or noisy columns out of AI clients platform-wide; the setting is global (System tenant), not per tenant.

Operational notes

- Tool calls consume ordinary web/API capacity and appear in API usage counters; there is no separate MCP usage meter.
- Since access rides on a normal user account, all the usual controls apply: disable the account to cut off a client, and review authentication activity in the audit trail.

9. Quality Assurance operations

9.1 Auto QA operations

Auto QA scores conversations against scorecards automatically. From the operator's perspective it is not a separate service: Auto QA is an AI task whose destination is **Auto QA**, executed by the *AI Assistant* job like every other task in the pipeline. This page covers how it runs and how to verify it per tenant; the tenant-side setup is in the Administration Guide.

How Auto QA runs

- An Auto QA task links a prompt to exactly one **evaluation form** (scorecard). The scorecard's questions and grading guidance are supplied to the AI automatically; the result is stored as a scored evaluation on the conversation.
- The task runs when the *AI Assistant* job processes the conversation — after transcription, and after the job's other destinations (Auto QA is deliberately last in the fixed destination order, so summaries, sentiment, and custom fields are already in place).
- **Licensing**: a conversation is auto-scored only if its assigned user holds the **Speech analytics license** (target name: Conversation Analytics license). The evaluation license gates *manual* evaluation work, not Auto QA.
- **Filters**: the task's filtering criteria (and the job's) decide which conversations are scored; conversations that already have an evaluation for the scorecard are skipped.
- Auto QA tasks can be **global with per-tenant overrides**, like any AI task: a shared global scoring task can be enabled, disabled, or re-prompted per tenant from the **Overrides** tab.

Auto QA consumes LLM tokens against the tenant's token limits — heavy scorecards on high volumes are usually the largest token consumer on the platform. Watch the **Usage** tab of the AI Tasks page (see AI tasks).

Verifying results per tenant

After enabling an Auto QA task for a tenant:

1. **Confirm the task state** — on **Administration › Conversation Analytics › AI Tasks**, check the tenant's rollup and the Overrides tab: is the task enabled for that tenant, and is the prompt Default or Overridden?
2. **Run one conversation through the playground** — the Auto QA form experiment in the AI Playground scores a real conversation of that tenant against the scorecard without persisting anything.
3. **Check the job** — the **Jobs** tab shows the AI Assistant job's runs, processing records, and logs; license errors and engine failures surface here (same workflow as transcription monitoring).
4. **Confirm evaluations appear** — in the tenant's scope, auto-scored evaluations show up under **QA › Evaluations** and on each conversation's QA tab as new conversations are processed. What supervisors see and how scores are calculated is documented in the User Guide QA chapter and Understanding Auto QA.

The legacy auto-score job

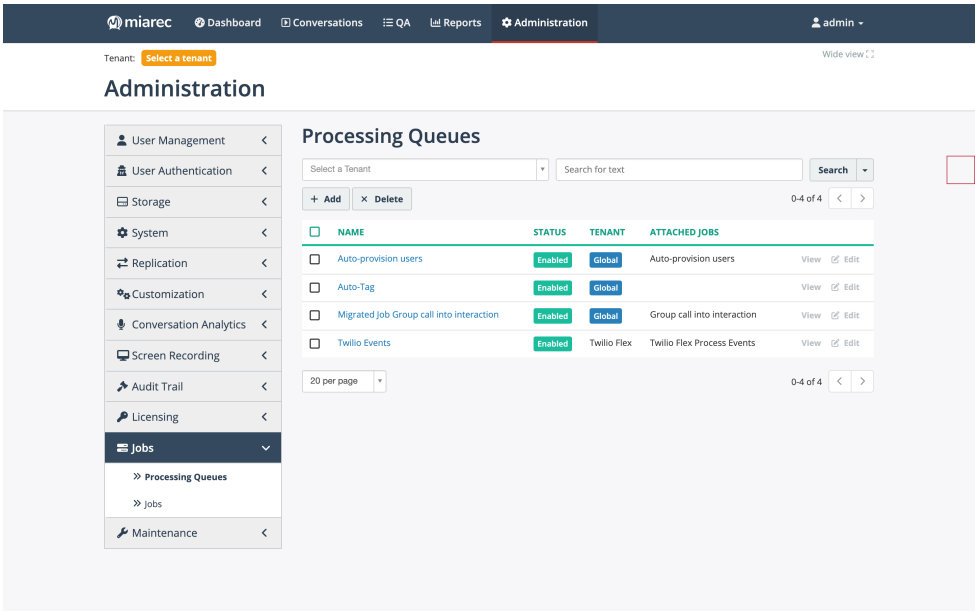
The jobs catalog still lists a standalone job type named **Auto QA (legacy)**. It is retired: it predates the AI-task pipeline and is not how Auto QA runs today. Do not create new instances of it — configure Auto QA as an AI task with the Auto QA destination, as described above.

10. Jobs & processing pipeline

10.1 Pipeline and queues

Processing queues are the glue between the stages of the processing pipeline. A queue collects records (usually conversations) when configured events fire — for example "recording finished" — and jobs attached to the queue consume those records, process them, and can publish them onward to the next queue. Chaining queues and jobs this way is how a conversation flows from capture through transcription, AI analysis, and delivery without any manual intervention.

Queues are managed under **Administration › Jobs › Processing Queues**; the jobs that read from and write to them are managed in the jobs console.



The Processing Queues list. Each queue shows its status, its scope (a Global badge or the owning tenant), and the jobs attached to it. A queue with no attached jobs — like "Auto-Tag" here — collects records that nothing consumes.

How a queue works

A queue definition has three parts:

- **Scope** — **Global** queues collect matching records from every tenant; tenant-scoped queues collect records from one tenant only. A global queue combined with job-level filters is the common pattern: the queue collects everything, the consuming job filters out records it should not process (for example, conversations of users without the required license).
- **Record type** — what kind of records the queue holds: **Call** (conversations) or **Integration events** (events pushed by a telephony integration, for example Twilio Flex).
- **Source events** ("Populate queue with events") — the platform events that add a record to the queue.

Source events for conversation queues:

Event	Fires when
Call - Recording started	A recording begins
Call - Recording finished	A recording ends — the usual trigger for transcription
Call - Created / Updated / Deleted	A conversation record is inserted, changed, or removed
Call - Transcrition completed	A transcript is ready — the usual trigger for AI tasks *(the label's typo is in the product)*
Call - Redaction completed	The redaction job finished with a conversation
Call - Upload completed	A manual or API upload finished
Call - Replication sent / received	Cross-site replication delivered a conversation
Integration event - Created	An integration pushed a new event (integration-events queues)

How jobs attach to queues

A job can be connected to a queue in three roles:

- **Source** — the job reads records from this queue and processes them. In the job definition this is the **Incremental using queue** data source with a selected source queue.
- **Destination** — the job publishes successfully processed records to this queue (**Publish to queue(s)** under "Action after successful processing").
- **Errors** — the job publishes records that failed processing to this queue, so failures can be retried or inspected separately.

A job whose source is a queue and whose schedule is **Continuously** processes new records as soon as they arrive — this is how the always-on pipeline stages run.

Example: the analytics pipeline

The standard Conversation Analytics wiring illustrates the pattern:

1. A global queue with record type **Call** and source event **Call - Recording finished** collects every finished recording.
2. The **Transcribe recordings** job consumes that queue continuously, filtered to users with a transcription license, and publishes completed records to a second queue.
3. The **AI Assistant** job consumes the second queue and runs the configured AI tasks on each transcript.

The full setup procedure is in Transcription system configuration and Pipeline overview.

> **Note**
>
> Nothing prevents a queue from existing without attached jobs — records simply accumulate. When you see a growing queue, check that a consuming job exists, is enabled, and is scheduled (ideally **Continuously**).

Related pages

- Jobs console — where jobs, their runs, and logs are managed.
- Job type reference — every job type, grouped by purpose.
- Pipeline overview — the Conversation Analytics pipeline end to end.

10.2 Jobs console

Jobs are persistent, scheduled or triggered background tasks: pipeline stages (transcription, AI tasks), storage chores (export, relocate, retention), telephony syncs, replication, and utilities. The jobs console at **Administration › Jobs › Jobs** is the single place where all of them are configured, run, and monitored.

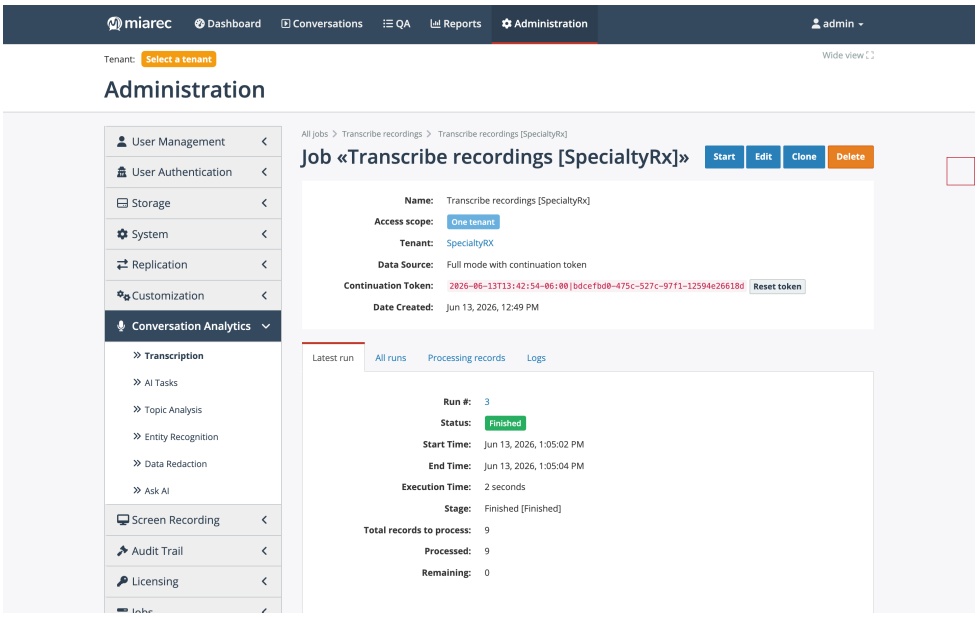
> **Note**
>
> Customer administrators see a reduced version of the same console with only the job types their role permits (tagging, confidential flag, telephony syncs) — see Jobs in the Administration Guide. Each job type is a separate row in the role's **Job permissions** section, so what a user sees here is entirely permission-driven.

Console tabs

- **Overview** — the job-type catalog: one row per job type with the number of configured jobs, their latest statuses, and the latest run. A tenant selector and text search narrow the list. Click a type to see its jobs, or **View** to jump to the latest run.
- **All jobs** — every configured job across all types, with name, tenant, status, schedule, and latest run.
- **Advanced Search** — find jobs by their attributes.

> **Warning**
>
> The Overview catalog does not list every registered job type. Some types are reachable only from their own menu section — for example the screen-recording export/import/relocate jobs live under **Administration › Screen Recording** — and internal types do not appear at all. Do not conclude from the catalog alone that a job type does not exist; see the job type reference.

The job page



A job page, Latest run tab. Note the header block — access scope, tenant, data source, and the continuation token with its Reset token button — and the Start/Edit/Clone/Delete actions.

The header block describes the job's definition:

- **Name** — free text; the convention `Type [Tenant]` keeps the All jobs list readable.
- **Access scope** — which data the job can touch: **Unrestricted** (all tenants, including System), **Tenants only** (all tenants, excluding System), or **One tenant** (with the tenant named below it).
- **Data source** — how the job selects records to process:
  - **Incremental using queue** — consumes records from a processing queue; the standard mode for continuous pipeline stages.
  - **Full mode with continuation token** — scans all matching records, remembering its position in a continuation token so the next run resumes where the last one stopped. **Reset token** makes the next run start over from the beginning.
  - **Full mode on each run** — rescans everything on every run.
  - **Incremental using replication slot** — consumes database change records; used by replication jobs.
  - **Incremental using DB storage (deprecated)** — a legacy incremental mode.
- **Date Created**.

The buttons **Start**, **Edit**, **Clone**, and **Delete** act on the job; a running job can be stopped with **Abort**. An inline audit trail at the bottom of the page records who changed or ran the job.

Schedules and notifications

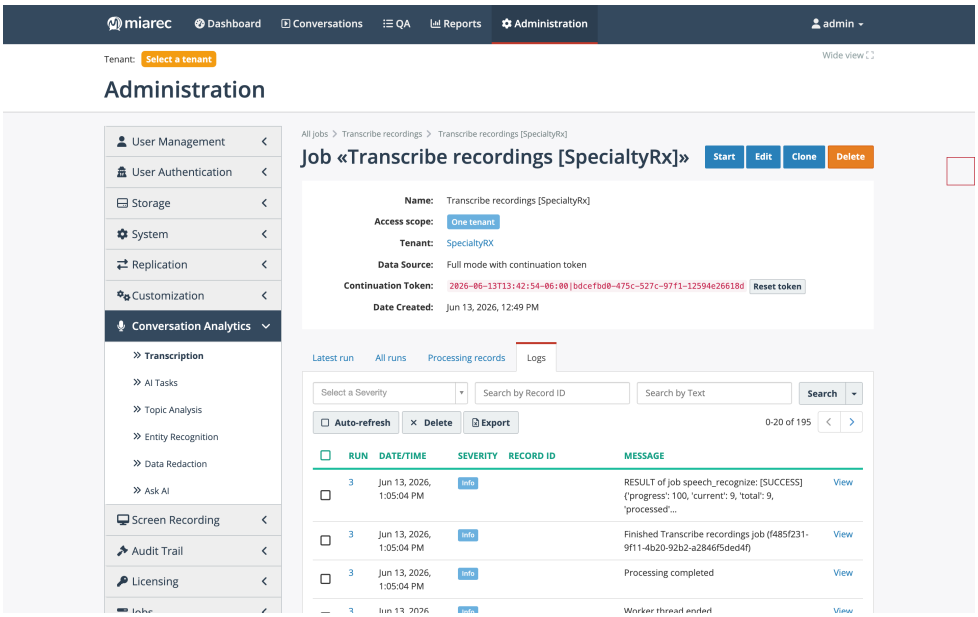
A job's **Schedule** section sets how it runs: **Not scheduled** (manual only), **Continuously** (keeps running and picks up new records as they arrive — for queue-fed jobs), **Every Hour / Day / Week / Month**, or a custom crontab expression. Job runs pass through statuses such as *Starting*, *In progress*, *Finished*, *Failed*, *Canceled*, and *Aborted*.

Jobs can email a notification on completion or failure — every time, only on the first of consecutive failures/completions, or never. Delivery uses the **Notification about job completion** and **Notification about job failure** email templates.

Runs, records, and logs

Each job page has four history tabs:

- **Latest run** — run number, status, start/end times, execution time, current stage, and record counters (total, processed, remaining).
- **All runs** — the run history with processed-per-day totals and per-run processed vs error counts, filterable by date range and status.
- **Processing records** — the individual records a run processed, searchable by record ID.
- **Logs** — the job's log entries with severity filter, record-ID and text search, auto-refresh, export, and per-entry delete.



The Logs tab of a transcription job: per-record messages with severity badges, plus Auto-refresh and Export. Filtering by Record ID is the fastest way to trace one conversation through a job.

Related pages

- Pipeline and queues — how jobs chain into a processing pipeline.
- Job type reference — what each job type does.
- Jobs — the customer-administrator view of the console.

10.3 Job type reference

Every job type registered on the platform, grouped by purpose. The jobs console Overview catalog presents these as one flat alphabetical list; this page is the map. Type names below are the exact labels used in the product (several still say "calls" or "recordings" — the labels predate the conversation-centric renaming).

> **Automatic Actions**
>
> Earlier releases exposed a dedicated **Automatic Actions** menu for the automation types (tag, confidential flag, group into interaction, auto-provision, signaling extraction). That menu is currently retired; the same job types are configured directly in the jobs console and are documented in the Automation group below.

Processing pipeline

The always-on stages that turn a captured conversation into transcripts and insights. They are normally fed by processing queues and scheduled **Continuously**.

Job type	What it does
Transcribe recordings	Sends recordings to a speech-to-text engine and stores transcripts. See Transcription system configuration.
AI Assistant	Runs the configured AI tasks (summary, sentiment, topics, custom insights, Auto QA) on transcribed conversations. See AI tasks.
Recognize entities	Runs entity recognition (NER) engines over transcripts. See Entity recognition.
Redact data	Applies data-redaction rules to audio and transcripts. See Data redaction.
Recalculate call metrics	Recomputes stored conversation metrics; used after configuration changes that affect derived values.
Resynchronize transcripts	Re-synchronizes stored transcript data for the selected conversations.
Auto QA (legacy)	*Legacy.* The standalone auto-scoring job from the pre-LLM era. Auto QA now runs as an AI task with the "Auto QA" destination — see Auto QA operations.
Extract topics/keywords (legacy)	*Legacy.* Keyword-spotting topic detection; topics are now extracted by the AI Assistant job.

Storage and data lifecycle

Bulk operations on stored recordings. Their dedicated pages in the Storage & data lifecycle chapter cover configuration in depth.

Job type	What it does
Export recordings	Copies recordings (with metadata) to a storage target for backup or handoff. See Export and backup.
Import call recordings	Ingests recordings produced by a third-party recorder. See Import recordings.
Bulk delete recordings	Retention: deletes data selectively by class (audio files, screen files, metadata) for conversations matching the job's own filters and schedule. See Retention and bulk delete.
Enforce storage limits	Purges the oldest data of tenants that exceed their storage quotas. See Storage limits.
Relocate audio files	Moves audio files between storage targets. See File maintenance jobs.
Convert audio files	Re-encodes stored audio to a different format.
Split audio by silence	Splits recordings at long silences.
Check files	Verifies that stored files exist and are readable, reporting discrepancies.

The do-not-delete flag

The **Enforce storage limits** job always skips conversations marked with the do-not-delete flag. The **Bulk delete recordings** job does **not** have a built-in exclusion — it deletes whatever its filters match, so retention jobs must be configured with the flag in mind. See Retention and bulk delete.

Replication

Cross-site replication jobs; setup is covered in Replication.

Job type	What it does
Replicate recordings	Sends conversations and their files to a replication target. See Replication jobs.
Replicate system configuration	Sends configuration data (tenants, users, settings) to a replication target.
Replicate data (legacy)	*Legacy, retired.* The old combined replication job; it refuses to run and points to the two jobs above.

Telephony synchronization

Integration jobs that pull conversations or events from a connected telephony platform. They are created per integration connection — see Integrations and the per-platform integration guides.

Job type	What it does
Twilio synchronize calls	Pulls conversations from a Twilio account.
Twilio Flex Process Events	Consumes Twilio Flex events from an integration-events queue and updates conversations.
Webex Native Recording synchronize calls	Pulls recordings from Webex.
NICE InContact synchronize calls	Pulls conversations from NICE InContact.
RingCentral Synchronize calls	*Hidden.* A RingCentral sync exists in the codebase but is not exposed in the catalog.

Automation

The former Automatic Actions: rule-driven changes applied to conversations as they arrive (queue-fed, continuous) or on a schedule. The first two are also available to customer administrators — see Jobs in the Administration Guide.

Job type	What it does
Tag calls	Assigns a tag to, or clears a tag from, conversations matching the job's condition.
Set confidential flag	Sets or clears the confidential flag on matching conversations.
Group calls into interactions	Links related conversation segments (for example transfers) into one grouped record.
Auto-provision users	Creates user accounts automatically upon a user's first conversation, using the tenant's auto-provisioning settings (group, role, licenses, login pattern). See Tenant configuration.
Extract signaling packet values	Extracts values from SIP/SIPREC signaling headers into custom fields.
Migrate interactions	*One-time migration utility* for converting legacy grouped-segment data to the current model.

Delivery

Job type	What it does
Send recordings via email	Emails matching conversations to a recipient list; attachments can include the audio file, transcript, metadata, and a PDF, using the recording email templates.
Send call metadata via webhook	Posts conversation metadata to an external webhook URL.

Screen recording

These three types are managed from **Administration › Screen Recording**, not from the jobs catalog — see Screen recording operations.

Job type	What it does
Export screen recordings	Copies screen-recording files to a storage target.
Import screen recordings	Ingests screen-recording files from a storage target.
Relocate screen recordings	Moves screen-recording files between storage targets.

Utilities and internal types

Job type	What it does
Publish/consume call queue	A diagnostic job that moves records between queues, with configurable delay and simulated errors; used to test queue wiring.
Generate random data / **Generate random evaluations**	*Internal test tooling* for seeding demo data; not for production use.

Related pages

- Jobs console — running and monitoring any of the above.
- Pipeline and queues — chaining jobs with processing queues.

11. Monitoring & maintenance

11.1 System log

The system log is the platform's error and event log: exceptions from the web portal, recorder service messages, integration failures, and other component-level events. It is stored in the database and viewed at **Administration › Maintenance › System Log**. Unlike the audit trail, which answers "who did what", the system log answers "what went wrong".

Reading the log

The **All System Log** tab lists entries newest first. Each entry has:

- **Severity** — Critical, Error, Warning, Info, or Debug.
- **Date** — when the event was recorded.
- **Source & Category** — the component and code location that produced the entry (for example `Web miarecweb.integrations.all.sso...` for a web-portal SSO error).
- **Message** — the event text; click through to see the full record, including the stack trace when one was captured.

Quick filters cover date range, severity, and source type; selected entries can be deleted. The **Advanced Search** tab combines criteria for more precise questions.

Retention

The **Retention Settings** tab controls automatic cleanup: **Delete old system logs** (Do not delete / Delete) and the retention period in days (minimum 30). With deletion off, the log grows until entries are deleted manually — on busy systems, set a retention period.

System log reports

Two report types turn the log into schedulable reports; they are created and run like any other report (see Global reports for operator specifics and the User Guide Reports chapter for the general workflow):

- **System Log Details Report** — one row per log entry. Available columns: ID, Date/Time, Date, Time, Source, Location, Severity, Category, and Message.
- **System Log Summary Report** — event counts (**Total events**) grouped by a time period: hour, day, week, month, or year. Use it to spot error spikes, then drill into the details report.

Related pages

- Log files — file-based logs of components that do not write to the system log.
- System status — current health of the host, database, and services.
- Audit trail — the user-activity counterpart.

11.2 System status

Administration › Maintenance › System Status is the live health page for the web-portal host and its backing services. Check it first when the portal is slow, storage is filling up, or background processing lags.

What the page shows

- **System** — OS version, hostname, system uptime, CPU count, and memory usage (with a usage bar).
- **Storage** — a per-device table: device, file path, file system, disk size, used space, and a disk-usage gauge. Watch the volume that holds the recordings directory.
- **Database** — PostgreSQL version, database uptime, total size, and the ten largest tables with approximate row counts. A **View database metrics** link opens the detailed database status pages — per-statement and per-connection views for diagnosing slow queries and connection pile-ups.
- **PGBouncer** — the connection pooler's version, or "Not detected" when PGBouncer is not in front of the database. A dedicated PGBouncer status page shows pool details when it is present.
- **Redis** — version, uptime, current and peak memory usage, and connected clients.
- **Task manager** — the **Tasks in queue** count of Celery tasks currently waiting. A persistently growing number means background processing (dashboards, reports, jobs) is falling behind.

Health-check endpoints

For external monitoring, the web portal exposes unauthenticated HTTP endpoints:

Endpoint	Purpose
`/health` and `/health/ready`	Readiness: checks PostgreSQL and Redis connectivity; returns HTTP 200 with `{"status": "ok", "postgresql": "...", "redis": "..."}` when both pass, HTTP 503 otherwise.
`/health/live`	Liveness: confirms the web process itself is responding.

Requests to `/health` and `/health/ready` are rate-limited (HTTP 429 when exceeded), so poll them at a moderate interval; `/health/live` is not rate-limited and is safe for frequent liveness probes.

Related pages

- System log — the error log to check when a gauge here looks wrong.
- Log files — where component logs live on disk.
- Maintenance tools — version information for every component.

11.3 Audit trail

The audit trail records who did what, across every tenant: sign-ins and sign-outs, playback and download of recordings, configuration changes, job and report activity, encryption-key grants, replication transfers, and more. As a root operator you see the cross-tenant view at **Administration › Audit Trail**; each tenant administrator sees the same feature scoped to their own organization — that customer-side view is documented in Audit trail in the Administration Guide.

The cross-tenant view

The **All Audit Log** tab lists records from all tenants, newest first. Each row shows the date, the **initiator** (the user who performed the action, with login), the **resource** affected, and the action with its details; **View** opens the full record including the changed data.

Filters on the list tab: a date range, a user-or-group picker, free-text **Search in DATA**, and **Filter by Resource** / **Filter by Action**. The **Advanced Search** tab adds precise criteria — including **Tenant** and **Tenant - ID**, which is how you narrow the cross-tenant log to one organization (or to records with no tenant association, i.e. system-level activity).

> **Inline audit trails**
>
> Most operator record pages — a tenant, a speech engine, a job — embed a collapsible audit-trail panel at the bottom showing only that record's history. It is often faster than filtering the global log.

Retention settings

The **Retention Settings** tab (also reachable at **Administration › Audit Trail › Retention**) controls automatic cleanup of audit records: **Delete old audit logs** (Do not delete / Delete) and the retention period in days, minimum 30. The default is unlimited retention — decide a policy deliberately, since audit records are compliance evidence for your tenants: the customer-side documentation tells administrators that retention "is configured by your service provider".

Audit reports

The **Audit Trail Details Report** and **Audit Trail Summary Report** types turn the log into schedulable, exportable reports. Their columns and filter semantics are documented in Audit reports; as an operator you can additionally create them as global reports or per tenant.

Related pages

- Audit trail — the customer-administrator view.
- System log — the component-error counterpart of the audit trail.
- Sessions and API usage — who is signed in right now.

11.4 Log files

The platform consists of several components, and not all of them write to the in-database system log. This page maps each component to where its log output lives.

Component	Log location
Recording service (MiaRec recorder)	Log messages in the database — **Administration › Maintenance › System Log**. With trace enabled, trace files are written to `/var/log/miarec/trace/` (Linux) or `Data\log\trace` (Windows).
Web portal	Log messages in the database (System Log). The Apache web server additionally logs to `/var/log/httpd/` (Linux) or `Data\log\apache` (Windows).
Celery (background task scheduler/workers)	`/var/log/miarecweb/celery/` or `/var/log/celery/` (Linux); `Data\log\celery` (Windows).
Redis (cache and task broker)	`/var/log/redis_*/` (Linux); `Data\log\redis` (Windows).
Operating system	`/var/log/messages` (Linux); Event Viewer (Windows).

Tip

Start with the System Log in the web UI — most recorder and web-portal errors surface there with stack traces. Fall back to the file logs when a component cannot reach the database at all (for example Celery failing to start, or Apache errors before the application loads).

Related pages

- System log — the in-database log and its retention settings.
- Troubleshooting — enabling recorder trace logging.
- System status — live health of the same components.

11.5 Troubleshooting

Administration › Maintenance › Troubleshooting hosts the recorder diagnostic tools. The one you will use in practice is the **Trace Log** — detailed trace output from the recording service, the primary evidence to collect when investigating recording problems.

Recorder trace log

The recording service can write detailed trace information to text files. Tracing is configured centrally from the Troubleshooting page: click **Edit Configuration** next to **Trace Log**. The settings are stored as recorder settings, so they apply to your recorder servers (per-server overrides are possible through the recorder settings tree — see Recorder servers).

Configuration options:

- **Enable** — turn trace-file writing on or off.
- **Trace log file name** — full path to the trace file. Defaults: `/var/log/miarec/trace/trace.log` on Linux, `Data\log\trace\trace.log` on Windows.
- **Trace level** — depth of trace information, 1 to 5. Use **5** (the default) when troubleshooting; it is the level MiaRec support expects in collected traces.
- **File rotation** — hourly, daily, weekly, or monthly rotation, with a rotate day and time where applicable.

The trace files themselves are read from the server's file system (see Log files); they are not viewable in the web portal.

> **Warning**
>
> Trace level 5 is verbose. Leave tracing enabled only while reproducing a problem, and keep rotation on so trace files do not fill the log volume.

A troubleshooting workflow

1. Reproduce the problem with tracing **enabled** at level 5.

2. Note the time and, if possible, the affected extension or phone number.
3. Collect the trace file(s) covering that window, plus any matching system log entries.
4. Attach both to your support case, or analyze the SIP/protocol exchange in the trace directly.

Related pages

- Log files — where every component logs.
- Recorder servers — the recorder registry and settings tree.
- System log — recorder errors surfaced in the web UI.

11.6 Maintenance tools

Two small pages under **Administration › Maintenance** round out the operator's toolkit: the UNDO list and the Version page.

UNDO list

Administration › Maintenance › UNDO List shows pending undo operations — typically delete operations that were performed with an undo grace period, such as deleted notes or deleted report history. Each entry shows the date/time, a description of what was deleted, and the user who initiated it, with an **UNDO** action to reverse the operation while the entry is still pending.

The list can be searched by description and filtered by user. When the list is empty, the page reads "UNDO list is empty".

> **Note**
>
> The UNDO list is a recovery convenience, not a backup. Once an entry is gone from the list, the deletion is final — restoring the data then is a database-restore exercise.

Version

Administration › Maintenance › Version collects the version of every component in one place:

- **Web portal** version.
- **PostgreSQL**, **Python**, **Redis**, and **OS** versions.
- A **recorder table** with the version of each running recorder, its service uptime ("Service started/stopped … ago"), or "Unknown" when the version cannot be determined. When no recorders are registered, the page shows "No recording servers detected".

Have this page open (or a screenshot of it) when contacting MiaRec support — component versions are the first thing a support case needs.

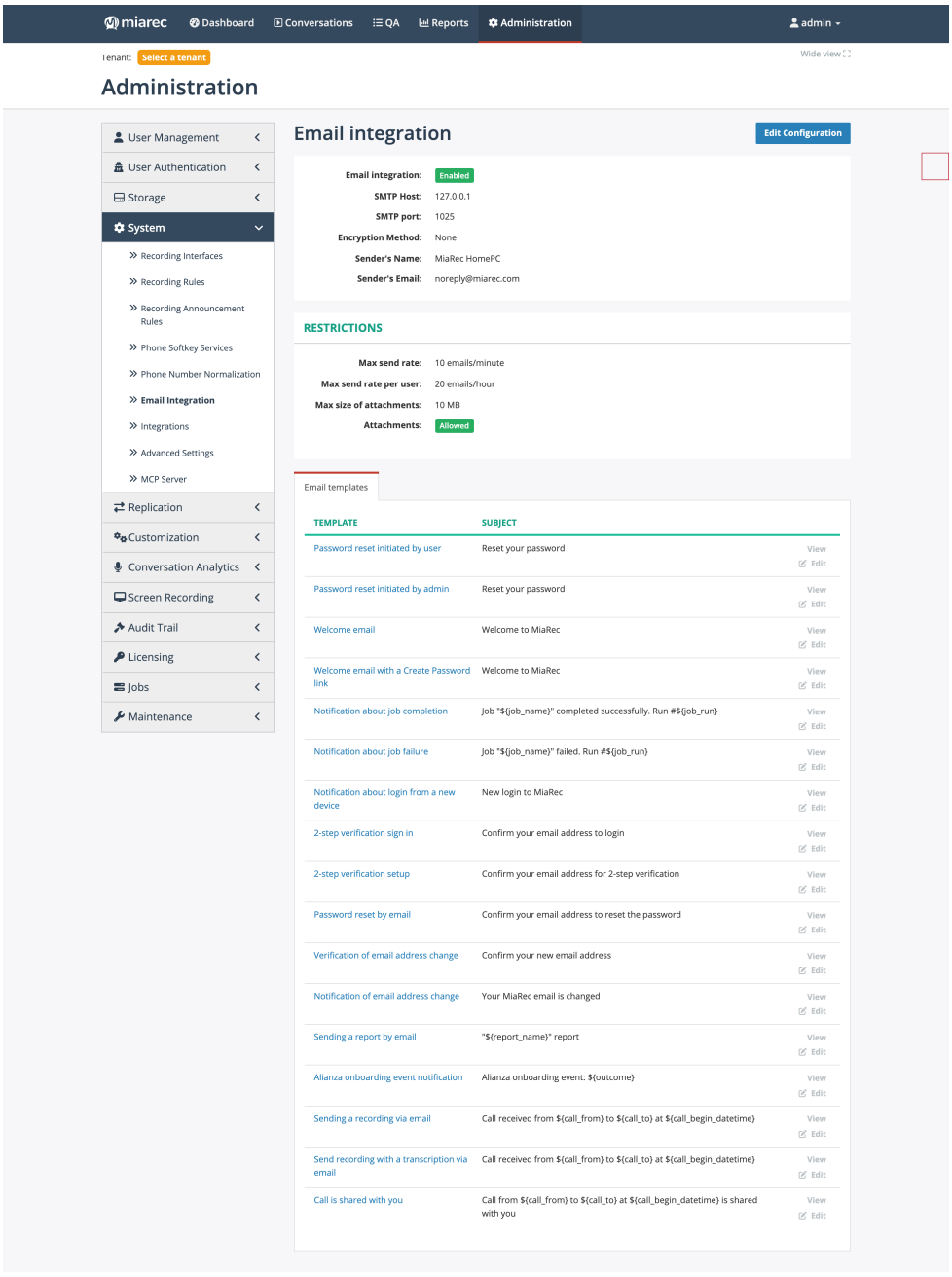
Related pages

- System status — live health for the same components.
- Recorder servers — the recorder registry the version table reads.

12. Platform services

12.1 Email

Everything the platform sends by email — password resets, welcome emails, 2FA codes, job notifications, report deliveries, shared conversations — goes through one system-wide SMTP integration, configured at **Administration › System › Email Integration**. The same page hosts the catalog of editable email templates.



The Email integration page: SMTP connection and sender at the top, sending restrictions in the middle, and the email template catalog below — every message the platform sends has a template row here with View and Edit actions.

SMTP configuration

Click **Edit Configuration** to set:

- **Enable Email integration** — the master switch; with it off, nothing is sent.
- **SMTP host** and **SMTP port** — usually 25 for non-encrypted connections, 465 for SSL, 587 for TLS/STARTTLS.
- **Encryption Method** and optional **authentication** (SMTP user and password).
- **Sender's Name** and **Sender's Email** — the From identity on all platform email.

The edit form includes a **Test Email Recipient** field and a **Test SMTP settings** button, so you can verify delivery before saving.

Restrictions

- **Max send rate** (emails/minute, system-wide) and **Max send rate per user** (emails/hour).
- **Max size of attachments** (MB) and an **Attachments** switch — whether users may attach recording and metadata files to an email at all.

Email template catalog

Each template defines the subject and body (plain text and HTML) of one message type, with `${parameter}` placeholders substituted at send time. **View** shows the template with its parameter

reference (name, description, example value); **Edit** lets you customize the wording, and a test send is available so you can check the result. The catalog:

Template	Sent when
Password reset initiated by user	A user requests a password reset
Password reset initiated by admin	An administrator triggers a reset for a user
Welcome email	A new user is created with a known password
Welcome email with a Create Password link	A new user must set their own password
Notification about job completion / about job failure	A job finishes, per the job's notification policy — see Jobs console
Notification about login from a new device	A user signs in from an unrecognized device
2-step verification sign in / setup	An email 2FA code is needed at sign-in or during enrollment
Password reset by email	A user confirms their address to reset a password
Verification of email address change / Notification of email address change	A user changes their email address
Sending a report by email	A scheduled or manual report delivery — see the User Guide Reports chapter
Alianza onboarding event notification	An Alianza integration onboarding event occurs
Sending a recording via email / Send recording with a transcription via email	The Send recordings via email job or a manual send delivers a conversation
Call is shared with you	A user shares a conversation with someone

The conversation-delivery templates expose conversation parameters (calling/called party name and number, begin and end date/time, duration, a link to the recording); the account templates expose user and link parameters. Placeholders must be kept as `${name}` — the parameter table on each template's page is the authoritative list for that template.

Related pages

- Two-step verification methods — email as a 2FA channel depends on this integration.
- Jobs console — job completion/failure notifications.

12.2 Localization

System-wide defaults for timezone and language live under **Administration › Customization**; individual tenants, groups, and users can override them. UI translations are maintained as gettext catalogs through a workflow with the MiaRec team.

Default timezone

By default the platform uses the timezone of the server it runs on. **Administration › Customization › Timezone** sets an explicit system default instead. Resolution is hierarchical: a user's own timezone wins, then their group's, then the tenant's, then this system default — the overrides are edited on the respective tenant, group, and user pages.

Timezone affects how dates and times are displayed throughout the portal, and reports additionally carry their own timezone setting for scheduling and date filters.

Default language

Administration › Customization › Language sets the default UI language ("Default" means English). Users pick their own language in their profile — see My profile in the User Guide — and tenants can carry a language of their own (visible as the "Tenant - Language" column in tenant reports).

The language list includes English, Spanish, French, German, Italian, Portuguese (Brazil), Russian, Bulgarian, and other European languages; translation coverage varies by language — languages translated through the workflow below display English text for any untranslated phrase.

Translation workflow

UI strings are internationalized with **gettext** (`.po` catalogs). To create or improve a translation:

1. Contact the MiaRec team and request the `.po` file for your language.
2. Translate it in POEdit or any editor supporting the gettext format.
3. Send the file back to MiaRec for inclusion in the distribution.

Two formatting rules matter when translating:

- Text in `${ }` brackets is a placeholder (for example `User ${name}`) — keep it exactly as is.
- Entries starting with `#` disambiguate identical English words with different meanings (for example `# From [date]` vs `# From [party]`) — translate the meaning, not the marker.

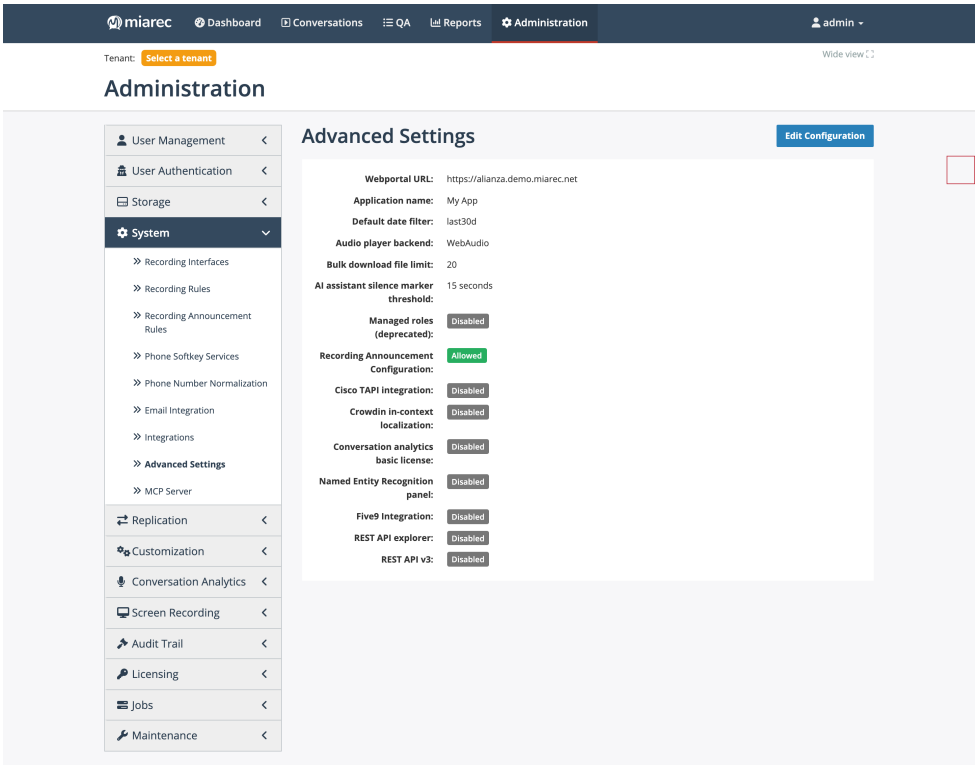
There is also a **Crowdin in-context localization** mode for translating directly in the running UI; it is gated by a feature flag (see Advanced settings) and used in coordination with the MiaRec team.

Related pages

- Advanced settings — the Crowdin flag and other system settings.
- Tenant configuration — per-tenant overrides.

12.3 Advanced settings

Administration › System › Advanced Settings collects system-wide behavior settings and the feature flags that show or hide optional functionality. Flags gate both menus and the matching permission rows in role forms, so a disabled feature disappears from the product entirely.



The Advanced Settings view page: general settings on top, feature flags below with Allowed/Disabled badges. Edit Configuration opens the form — which also contains a few flags not shown on this view page (Broadworks and Metaswitch integration).

General settings

Setting	Meaning
Webportal URL	The canonical base URL of the portal, used when building links in emails and integrations.
Application name	The product name shown in the UI and browser titles (white-labeling; default "MiaRec").
Default date filter	The initial date filter on list pages: Last 7 days, Last 30 days, or Disabled.
Audio player backend	WebAudio or MediaElement — which browser playback engine the players use.
Bulk download file limit	Maximum files per bulk ZIP download; default 20, configurable 1–1000.
AI assistant silence marker threshold	Minimum silence length (seconds, default 15) marked as a silence gap for the AI assistant.

Feature flags

Flag	Effect when allowed
Managed roles (deprecated)	Legacy managed-roles behavior; leave disabled.
Recording Announcement Configuration	Shows **System › Recording Announcement Rules** and the per-tenant announcement settings — see Announcement rules.
Cisco TAPI integration	Adds the Cisco TAPI recording interface (supported on Windows servers only).
Broadworks integration / Metaswitch integration	Enable the Broadworks and Metaswitch portal-authentication and association features — see Portal authentication. Set from the Edit Configuration form.
Crowdin in-context localization	Enables the in-UI translation mode — see Localization.
Conversation analytics basic license	Unlocks a deprecated per-user license sub-tier slated for removal; do not enable on new systems.
Named Entity Recognition panel	Shows the user-facing NER panel on conversation pages. The NER admin pages (engines, jobs) are visible regardless — see Entity recognition.
Five9 Integration	Shows the Five9 VoiceStream configuration menus — see Recording interfaces.
REST API explorer	Enables the interactive API explorer UI.
REST API v3	Enables the v3 REST API (returns 403 when off). The APIs themselves are documented in the REST API guide.

> **Note**
>
> Some functionality is gated by build/menu availability rather than by a flag on this page — if a menu item from this guide is missing and no flag matches, verify the portal build before assuming misconfiguration.

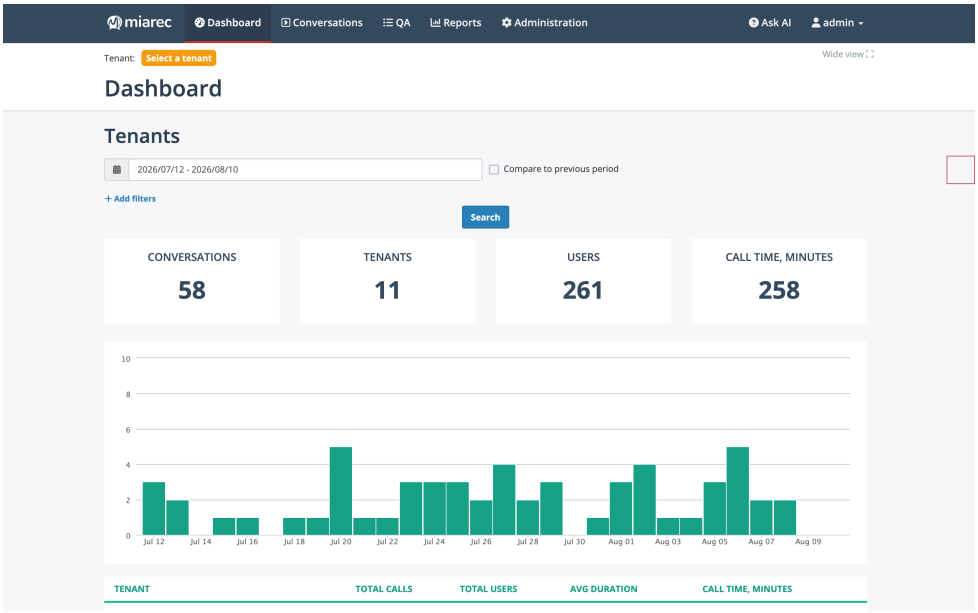
Related pages

- Localization
- Recording interfaces
- REST API guide

13. Reporting on tenants

13.1 Cross-tenant dashboard

Users with cross-tenant visibility land on the **Tenants** dashboard after signing in — an aggregate view of activity across every tenant, at **Dashboard** (`/dashboard/tenants/overview`). It is the only operator-specific dashboard; drilling into a tenant opens the same analytics dashboards that tenant's own administrators see.



The Tenants dashboard: KPI tiles for Conversations, Tenants, Users, and Call time; the daily conversation trend; and (below) the per-tenant table. The date filter and "Compare to previous period" apply to the whole page.

What the page shows

- **KPI tiles** — **Conversations**, **Tenants**, **Users**, and **Call time, minutes** for the selected period. With **Compare to previous period** checked, the conversation and call-time tiles show the change against the preceding period of the same length.
- **Trend chart** — conversations per day across all tenants.
- **Per-tenant table** — one row per tenant with **Total calls**, **Total users**, **Avg duration**, and **Call time, minutes**. Click a tenant to drill into its dashboard.

The date filter defaults to the system's default date filter (see Advanced settings); **Add filters** narrows the data with the same advanced filters used elsewhere.

Panels are computed by background tasks and cached for a few minutes, so tiles can briefly show a loading state and numbers can lag real time slightly.

Drilling down

Selecting a tenant opens that tenant's analytics dashboard — overview, sentiment, sentiment heatmap, topics, and conversations tabs — described in the User Guide dashboards chapter. The QA dashboard has a matching cross-tenant entry point that lists tenants with their evaluation totals before drilling into a tenant's QA overview.

> **Tip**
>
> For numbers you need as a document rather than a page — per-tenant call volumes, storage usage — use the tenant reports instead; they export to XLSX/PDF and can include storage and license columns the dashboard does not show.

Related pages

- Tenant reports — the reporting counterpart of this dashboard.
- View-as and impersonation — inspecting a tenant's data beyond dashboards.

13.2 Global reports

Report templates have three visibility scopes. Two of them exist inside a tenant — **private** (owned by one user) and **public** (shared with the tenant's report users). The third is operator-only: a **global** report belongs to no tenant and no owner, and is visible in *every* tenant. Global reports are how you publish a standard report — a monthly usage summary, a compliance extract — to all your tenants at once.

The general reporting workflow (layout, filters, charts, running, exporting) is the same as for any report and is documented in the User Guide Reports chapter; this page covers only the operator-specific parts.

Creating a global report

When a cross-tenant operator clicks **Create** under **Reports**, the create dialog asks for the target scope before anything else:

- **Visibility** — **Global (visible to all tenants)** or **Tenant (visible to a single tenant)**.
- **Tenant** — the target tenant, shown when "Tenant" is selected.

Choosing **Global** creates the tenantless, ownerless global report; choosing **Tenant** drops you into that tenant's normal report form, where the in-tenant Public/Private visibility and Owner fields apply. The same select-tenant dialog appears when cloning or importing a report template.

Creating a global report requires a system-tenant administrator with tenant visibility and the public-reports add permission. Without the global-report permission, the report is created as a private report in the System tenant instead.

What is different about global reports

- **No schedule** — global reports cannot be scheduled or emailed on a schedule; only private (owned) reports can. To schedule a standard report for one tenant, clone the global template into that tenant as a private report with an owner.
- **No owner controls** — the Owner and in-tenant Visibility fields are hidden; there is nothing to set.

- **Results are per-runner** — as with public reports, the data a run returns is scoped by the permissions of the user who runs it. A tenant administrator running your global report sees only their organization's data; you running it from the System tenant see everything your role allows.
- **Report-type availability still applies** — operator-only types (for example the tenant reports) never become visible to tenant users just because the template is global; tenants only see global reports of types their access level permits.

Related pages

- Tenant reports — the operator-only report types.
- Report templates and Running reports — the general workflow.

13.3 Tenant reports

Two report types exist specifically for reporting *on* tenants: the **Calls Summary Report by Tenants** (activity per tenant) and the **Tenant Details Report** (configuration, licensing, and storage per tenant). Both are available only to cross-tenant roles working from the System tenant, and both export to XLSX and PDF. The general reporting workflow is in the User Guide Reports chapter.

> **Note**
>
> Both types also work on a non-multitenant installation — the Tenant Details Report is the standard way to get storage usage numbers even with a single organization.

Calls Summary Report by Tenants

One summary row per tenant for the selected period, with drill-down (**View calls**) into the underlying conversations. Columns combine three groups:

- **Tenant - Name** (default) and **Tenant - ID** (optional).
- The full call-summary column library — conversation counts and durations by direction, per-weekday counts, hold/queue/talk time, sentiment buckets, and so on — documented in the User Guide column reference.
- Storage usage and limit columns. The usage columns count the conversations matching the report's period and filters (not tenant-wide totals — for those, use the Tenant Details Report):

Column	Description
Storage Usage - Audio Recordings - File Size, GB	Audio file size of the matching conversations
Storage Usage - Audio Recordings - Minutes	Audio minutes of the matching conversations
Storage Usage - Total Audio Recordings	Number of matching audio recordings
Storage Usage - Screen Recordings - File Size, GB	Screen-recording file size of the matching conversations
Storage Usage - Screen Recordings - Minutes	Screen-recording minutes of the matching conversations
Storage Usage - Total Screen Recordings	Number of matching screen recordings

Matching **Storage Limits** columns are also available: Enabled; Audio Minutes / File Size, GB / Days; Screen Minutes / File Size, GB / Days.

Tenant Details Report

One row per tenant, reflecting the tenant's *current* configuration — the report has no report period and no caching, and its rows drill down (**View**) into the tenant page. It is the reporting counterpart of tenant configuration. Columns come in three groups.

Tenant profile and settings columns

Column	Description
Tenant - Name / Tenant - ID	Identity
Tenant - Language / Tenant - Timezone / Tenant - Domain	Localization and branding-domain settings
Tenant - Total Users	All user accounts in the tenant
Tenant - Total Recorded Users	Users whose conversations are captured
Tenant - Total Users (filtered)	User count under the report's filters
Tenant - Extension Type	How extensions are interpreted for the tenant
Tenant - Extensions Uniqueness	"System wide" or "Within tenant only"
Tenant - Associate Calls	The conversation-to-tenant association method
Tenant - Associate Calls with Tenant - Broadworks SP ID / Broadworks Group ID / MetaSwitch System Name / MetaSwitch Group Name / Cisco Partition / SIP URI Host / Condition	The platform-specific association values
Tenant - Record Unknown Users	Whether conversations of unknown users are kept
Tenant - User Auto Provisioning (+ Web Portal Access, Group, Role, Recording Settings, Login Match Pattern, Login Translation Rule, Authentication Type, Licenses)	The auto-provisioning policy and its details
Tenant - Screen Recording Setting	Screen recording enabled/disabled
Tenant - File Format / Tenant - Audio Format / Tenant - AGC Filter / Tenant - PLC Filter	Recorder audio settings (Stereo/Mono, filters), "Default" when inherited
Tenant - LookBack On-Demand Recording / Tenant - Pause Recording Timeout	On-demand and pause behavior
Tenant - Transcription Prompt	The tenant's transcription vocabulary prompt

License columns

Per-tenant license limits, as configured on the tenant page (see License usage for how limits, keys, and assignment relate):

Column	License
Tenant - License - Call Recording	Recording seats
Tenant - License - Screen Recording	Screen-recording seats
Tenant - License - Live Monitoring	Monitoring seats
Tenant - License - Agent Evaluation	QA (evaluation) seats — legacy label
Tenant - License - Conversation Analytics	Conversation Analytics seats

Storage limits and usage columns

Column	Description
Storage Limits - Enabled	Whether storage limits are enabled for the tenant
Storage Limits - Audio Recordings - Minutes / File Size, GB / Days	The audio quota by minutes, size, and age
Storage Limits - Screen Recordings - Minutes / File Size, GB / Days	The screen-recording quota
Storage Usage - Audio Recordings - File Size, GB / File Size, %	Audio storage used, absolute and as % of the limit
Storage Usage - Audio Recordings - Minutes / Minutes, %	Audio minutes used, absolute and as % of the limit
Storage Usage - Total Audio Recordings	Number of audio recordings
Storage Usage - Screen Recordings - File Size, GB / File Size, %	Screen storage used
Storage Usage - Screen Recordings - Minutes / Minutes, %	Screen minutes used
Storage Usage - Total Screen Recordings	Number of screen recordings

The percentage columns are the quickest way to spot tenants approaching their quotas before the Enforce storage limits job starts purging.

Other operator-oriented report types

- **System Log Details / Summary Reports** — see System log.
- **Audit Trail Details / Summary Reports** — see Audit trail.
- **User Details Report** — includes operator-only columns (per-user licenses, managed tenants) beyond the customer-visible set documented in User reports.

Related pages

- Cross-tenant dashboard — the interactive counterpart.
- Global reports — publishing standard reports to all tenants.
- Storage limits — configuring the quotas these columns report on.

14. Screen recording operations

14.1 Screen recording operations

Screen recording captures the agent's desktop alongside the audio of a conversation. The operator-side pieces live under **Administration › Screen Recording**: system settings, the recordings list, workstation authorization, session monitoring, the screen-recording server registry, and three file-maintenance jobs. Client installation and end-to-end setup are covered by the dedicated Screen Recording Administration Guide; the customer-facing summary is Screen recording in the Administration Guide.

Screen Recording Settings

Screen Recording Settings holds the system-wide configuration:

- **Controller** — the TCP port clients connect to, an optional TLS port with SSL key/certificate/CA files, and client trace-log settings.
- **Capture** — capture rate (0.25–8 fps), multi-monitor support (primary monitor only or all monitors), maximum image width/height, and video bitrate.
- **Recording behavior** — wrap-up time (how long capture continues after the conversation ends) and maximum recording duration.
- **Upload** — the upload recordings URL clients post to, the storage target, the video file name format, and the upload chunk size.

Licensing

Screen recording is licensed separately:

- The **Screen Recording License** page (**Administration › Licensing › Screen Recording License**) shows the license and the registered screen-recording module instances (instance, location, version, status) — see License keys.
- Per-tenant screen recording is switched on the tenant page, and SCR seats are assigned per user; usage appears in License usage.

Day-to-day operations

- **Screen Recordings** — the cross-tenant list of captured screen recordings, with advanced search.
- **Client Workstations** — the registry of capture clients with authorization filters **Authorized**, **Pending**, and **Forbidden**; new workstations wait in Pending until you **Authorize** (or **Forbid**) them. Deleting a workstation removes its registration.
- **Client Sessions** — current and past client connections, for verifying that a workstation is online and uploading.
- **Recording Servers** — the screen-recording module registry (the servers that receive and process uploads), with per-server settings views and overrides, like the audio recorder servers registry.

File-maintenance jobs

Three job types manage screen-recording files; they are reached from the Screen Recording menu (not the jobs catalog) but behave like any other job in the jobs console:

Job	Purpose
Export Screen Recordings	Copy screen-recording files to a storage target for backup or handoff.
Import Screen Recordings	Ingest screen-recording files from a storage target.
Relocate Screen Recordings	Move files between storage targets, for example onto cheaper storage.

Screen recordings are also covered by per-tenant storage limits (screen size/minutes/days quotas) and retention — see Storage limits and Retention and bulk delete.

Related pages

- Screen Recording Administration Guide — client rollout and full setup.
- License usage — SCR seat assignment per tenant and group.
- Job type reference — the three job types above.

