Comma

ADMIN GUIDE

Set Up RCS/SMS Archiving on Android with Google MDM

Deploy the Comma Compliance Messages Archiver to fully-managed Android devices with Google Workspace Endpoint Management

Comma Compliance · 2026-06-30

ADMIN GUIDE

Set Up RCS/SMS Archiving on Android with Google MDM

Deploy the Comma Compliance Messages Archiver to fully-managed Android devices with Google Workspace Endpoint Management

Who this guide is for

You are a Google Workspace administrator and you want to archive your team's **RCS, SMS, and MMS** messages from corporate Android phones for compliance. This guide walks through deploying the **Comma Compliance Messages Archiver** (`com.commacompliance.archiver`) using Google's own tooling - **Google Workspace Endpoint Management**, **managed Google Play**, and **managed configuration** from the Google Admin console (admin.google.com). The console handles app approval, auto-install, and the archiver's own configuration; one undocumented Google Messages key may require a managed-config path that supports raw key/value (see Step 3b).

The archiver runs on **fully-managed (device-owner) Android Enterprise devices**. It reads the SMS/MMS/RCS messages that Google Messages stores on the device and ships them - encrypted - to your Comma backend. It does not intercept anything in transit, and while it is active the phone shows a **persistent "archiving for compliance" notification** naming your organization, so capture is never hidden from the user.

> **WHAT IS AND ISN'T COVERED**
>
> This archiver captures RCS, SMS, and MMS that flow through the phone's default messaging app. WhatsApp, Signal, Telegram, and iMessage are separate channels with their own Comma connectors - they are not part of this Android setup.

Where to get the app

The archiver is the **Comma Compliance Messages Archiver** on Google Play. For a managed fleet you bring it in through **managed Google Play**: Comma makes it available to your enterprise's managed Google Play - you approve the listing in your Admin console, or Comma targets the app privately to your organization - and your MDM force-installs it (Step 1). The same app is also published on the **public Play**

Store (`https://play.google.com/store/apps/details?id=com.commacompliance.archiver`), which is the right way to install it on a one-off or self-serve device. Either way, installing the app does not by itself start capture: an admin must have enabled Android RCS+SMS capture for the team, the device must be fully-managed, and Google Messages must be wired to the archiver with the `messages_archival` managed-config key (Step 3b) - so even a self-serve install relies on your MDM to deliver that key.

What you'll need

Have these values ready before you start. You'll enter them in the Comma web app and in the Google Admin console.

Value	What it is	Example / where to get it
Organization / backend URL	Your Comma backend address	`https://app.commacompliance.com` (self-hosted orgs use their own URL; must be `https`)
Enrollment token	The managed-enrollment credential - **token paths only**	Request from Comma. A **shared fleet token** (reusable across the fleet) or a **single-use device token** (one phone). Optional expiry 1-90 days (default 7). Begins `amenr_`
Backfill days	Days of existing messages to capture on first run	`90` (`0` = forward-only)
Archiver app package	The app you are deploying	`com.commacompliance.archiver`
Google Messages package	The default messaging app you wire to the archiver	`com.google.android.apps.messaging`

How a device enrolls: three options

A managed device has to do two things: (1) be told **which app Google Messages should wake** when a message arrives, and (2) **enroll with your Comma backend**. Step 1 (the `messages_archival` key) is the same for everyone. For step 2, choose one of three enrollment options:

- **Shared fleet token (recommended for fleets).** Push **one reusable token** to every phone via managed configuration. Each phone enrolls itself on first run and claims a seat - **no user interaction**

and no per-device setup. Optionally cap seats or set the token's expiry (1-90 days, default 7). This is the lowest-touch path for company-owned fleets, and the worked example below uses it.

- **Single-use device token.** A one-time token issued for **a single phone**; it binds to that device on first enrollment and is managed individually. Useful for enrolling one device at a time.
- **OAuth self-service.** No token - the employee signs in once in the app and the device registers under their team membership. Best when employees set up their own corporate phones. It asks each user to sign in per device, so it is the heaviest path for a large rollout.

This guide covers the **shared fleet token** as the worked example, then the OAuth self-service alternative.

Before you start: enable capture in the Comma web app

These steps happen **once, in the Comma web app**, before any device connects. An administrator must do them - a user cannot self-enable archiving.

1. Sign in to the Comma Compliance web app (https://app.commacompliance.com).
2. Go to **Account -> your team -> Android RCS+SMS capture**.
3. Turn on **"Enable Android RCS+SMS capture for this organization."** Until this is on, no device can enroll, OAuth sign-in is refused with a plain-language message, and no message upload is accepted.
4. Set **"Default backfill window (days)"** - for example `90` (`0` = forward-only). Saving this updates your team's **already-connected** Android devices; it is not a default that devices connecting later inherit. For the token paths, set `backfill_days` per device in managed config (Step 3a) - that is the explicit, reliable control.
5. **Token paths only:** generate your **enrollment token** in the same screen. A **shared fleet token** is reusable across the fleet (with an optional seat cap and an expiry of 1-90 days, default 7); a **single-use device token** binds to one phone. The token is shown once and is not stored in cleartext, so copy it when it appears. *OAuth self-service deployments skip this step entirely.*

> **WHAT YOU DO NOT CONFIGURE**
>
> The OAuth client id (comma-android-messages-archiver), the sign-in and token endpoints, and the device-integrity mode are auto-discovered by the app from your backend (GET /api/v1/android_archiver/config). You never set these by hand.

Step 1 - Approve the app in managed Google Play

1. In the **Google Admin console** (admin.google.com), go to **Apps -> Web and mobile apps**.
2. Click **Add app**. Comma makes the **Comma Compliance Messages Archiver** available to your enterprise's managed Google Play - search for it, or approve the public listing (`https://play.google.com/store/apps/details?id=com.commacompliance.archiver`). If it doesn't appear, contact Comma (`contact@commacompliance.com`) to have it targeted to your organization.
3. Select the app and **Approve** it. If prompted, accept the requested permissions on behalf of your organization. The app declares the restricted `READ_SMS` permission as a permitted **enterprise archiving** use case under Google Play's SMS and Call Log Permissions policy - it reads the messages your default messaging app stores; it does not send or modify messages.

Step 2 - Auto-install to fully-managed devices

1. Open the app in **Apps -> Web and mobile apps** and edit its **Settings**.
2. Choose the **organizational unit (or group)** that contains your fully-managed Android devices.
3. Set **User access** to **ON** and turn on **Auto-install** (force install) so the app deploys to those devices with no user action.

DEVICE REQUIREMENTS

- **Fully-managed (device-owner)** Android Enterprise device - enrolled as **company-owned**. Work profile, BYOD, and COPE are **not** supported.
- **Google Messages set as the default SMS/RCS app**, with RCS chat features provisioned. On phones where another app (e.g. Samsung Messages) is the default, switch the default handler to Google Messages.
- A **current Google Messages build.** The capture trigger only fires on a recent build - allow or push the Play update for `com.google.android.apps.messaging` . An out-of-date Messages app captures nothing, silently.
- **Protect the apps from removal.** Add both `com.commacompliance.archiver` and `com.google.android.apps.messaging` to your Android **uninstall / user-control restrictions** so a user can't force-stop, disable, or clear them and break capture.

Step 3 - Apply the managed configuration

This is the heart of the setup. You apply managed configuration to **two** apps.

3a. Configure the archiver app

In **Apps -> Web and mobile apps -> Comma Compliance Messages Archiver -> Managed configurations**, add a configuration with these keys (they are published in the app's schema, so they appear directly in the editor), and assign it to the same org unit:

Key	Type	Value
`backend_url`	string	`https://app.commacompliance.com` (your backend; must be `https`)
`enrollment_token`	string	Your **shared fleet token** (`amenr_...`) - the same token enrolls every phone in the OU. Use a **single-use device token** only when assigning to one device. **Omit for OAuth self-service.**
`backfill_days`	integer	`90` (optional; `0` = forward-only). If omitted, the value returned at enrollment applies

Because the fleet token is reusable, **one configuration enrolls the whole OU** - there is no per-device work. (A single-use token is the exception: assign it to one device only.)

> **GO FULLY UNATTENDED: PRE-GRANT THE SMS PERMISSION**
>
> On fully-managed devices, many MDMs can pre-grant runtime permissions to a managed app. Where your MDM supports it, grant the message-access (SMS) permission to the archiver in your app permission policy so the user isn't prompted on first launch - verify it in your environment, since otherwise the app asks for message access once. Combined with a shared fleet token, enrollment then completes with no sign-in and no per-device step.

3b. Wire Google Messages to the archiver

Google Messages needs to know which app to wake on each message event. Add (or open) the **Google Messages** app (`com.google.android.apps.messaging`) in **Web and mobile apps**, then add a managed configuration with this single key:

Key	Type	Value
`messages_archival`	string	`com.commacompliance.archiver`

IMPORTANT: THE

messages_archival key may not appear in the console editor messages_archival is an undocumented key that Google Messages does not publish in its configuration schema. The Google Admin console's managed-configuration editor is schema-driven , so it only lists published keys and may not let you add this one. This key is required on every enrollment path - OAuth self-service only removes the enrollment_token , not the Google Messages wiring. If your console can't add it:

- Deploy through a third-party EMM (Microsoft Intune, VMware Workspace ONE) or the **Android Management API**, all of which support **raw key/value** managed-config bundles where this key can be set directly.
- If none of those are available, contact Comma (`contact@commacompliance.com`) - capture depends on this key.

Step 4 - Verify on the device

1. The archiver installs automatically. Once it enrolls, the phone shows a **persistent "archiving for compliance" notification** that names your organization. This notification can't be dismissed while archiving is active.
2. Open the app - it shows it is **Connected** and archiving for your organization.
3. Send a **test SMS** and a **test RCS** message to the phone. Within a short time they should appear in your Comma archive under **Messages -> RCS**.
4. In the Comma web app, the device appears under **Account -> your team -> Messages Archiver devices** with an **Active** status and recent **"Last seen" / "Last upload"** timestamps. You can revoke any single device here, or rotate / revoke the fleet token to stop new enrollments.

If nothing arrives, see **Troubleshooting** below.

How enrollment is secured

Whichever option you choose, enrollment is protected the same way:

- **Signed two-step handshake.** The device generates an **Ed25519** key pair, the server issues a one-time random challenge, and the device proves possession of its private key by signing it. The **private key never leaves the device.**
- **Device attestation (Google Play Integrity).** The server can verify the device and app are genuine before allowing archiving (configurable).

- **One-time and time-boxed.** Each challenge expires in about five minutes and can be completed only once - no replay.
- **Revocable.** Revoke a device and it stops archiving immediately; rotating or revoking a token prevents further enrollments with that token.

After enrollment, each device holds **its own archiving credential** for its team's Android archive, and uploads are accepted only for enrolled device IDs under that team.

The OAuth self-service alternative

If you'd rather have employees connect their own corporate phones, use OAuth self-service. You still complete the Comma web-app prerequisite (enable **Android RCS+SMS capture** for the team), you still wire Google Messages with `messages_archival`, and you can still auto-install the app via managed Google Play; you just **omit** the `enrollment_token`.

The employee then:

1. Opens the app. On first launch it shows a plain-language summary of what's captured and, unless message access is already granted by device policy, asks for the **message-access (SMS) permission** - the employee grants it.
2. Taps **"Connect to your organization"** and enters the organization URL: `https://app.commacompliance.com`.
3. Signs in with the usual work account when the **secure browser tab** opens (OAuth in a Chrome Custom Tab - no password is ever typed into the app).
4. Taps **Authorize**.

The device registers under that employee's own membership and begins archiving - the same persistent notification appears. Self-service works only if an administrator has enabled **Android RCS+SMS capture** for the team; otherwise sign-in is refused with a clear "not enabled yet" message.

Troubleshooting

Symptom	Most likely cause	Fix
No messages arrive at all	Out-of-date Google Messages build (the capture trigger never fires)	Push the Play update for `com.google.android.apps.messaging`
No messages arrive, build is current	`messages_archival` missing or mistyped on Google Messages	Re-check the key/value; if the Admin console won't accept the key, use a raw-key EMM or the Android Management API (see Step 3b)
OAuth self-service sign-in is refused	"Android RCS+SMS capture" not enabled for the team	An admin enables it in the Comma web app
Some phones don't enroll on the fleet token	Seat cap reached, or the token expired or was rotated	Raise or clear the seat cap, or issue a fresh token, in the Comma web app
Capture stops after a user action	User force-stopped or disabled the app	Add both packages to uninstall / user-control restrictions
Device never appears as connected	Device isn't fully-managed (device-owner)	Re-enroll the device as company-owned (fully managed)

Reference: configuration requirements

On the archiver app (`com.commacompliance.archiver`):

Key	Type	Required	Value
`backend_url`	string	Yes (all paths)	Your Comma backend, `https://...`
`enrollment_token`	string	Token paths only	Shared fleet token (reusable) or single-use device token; omit for OAuth self-service
`backfill_days`	integer	Optional	Days of history on first run; `0` = forward-only

On Google Messages (`com.google.android.apps.messaging`):

Key	Type	Required	Value
`messages_archival`	string	Required (all paths)	`com.commacompliance.archiver`

Auto-discovered by the app (you do not set these): OAuth client id `comma-android-messages-archiver` , authorization/token endpoints, and integrity mode - all served by your backend at `GET /api/v1/android_archiver/config` .

> **NEED HELP?**
>
> Email contact@commacompliance.com to have the app targeted to your organization's managed Google Play or to request enrollment tokens. Learn more at commacompliance.com.