Comma

ARCHIVE EXPORT - AZURE SETUP

Deliver Your Archive to Azure Blob

Step-by-step setup for delivering your Comma compliance archive into a customer-owned Azure Blob container

Comma Compliance · 2026-07-06

ARCHIVE EXPORT - AZURE SETUP

Deliver Your Archive to Azure Blob

Step-by-step setup for delivering your Comma compliance archive into a customer-owned Azure Blob container

This guide covers everything you provision on the **Azure Blob Storage** side so Comma can deliver your archive into a container you own.

Delivery authenticates with a **storage account key you hand to Comma**. It performs block-blob **writes** for delivery, and the one-time Verify & activate probe additionally **reads back** and **deletes** a single tiny object - so the key needs write + read + delete on the container.

> **SAS TOKENS ARE NOT SUPPORTED IN V1**
>
> Comma's Azure client authenticates with a storage account name plus an account key (or a managed identity); it cannot authenticate from a caller-supplied SAS token. Provide an account key . Because an account key grants the whole account, scope it by dedicating a storage account to Comma delivery - do not reuse an account that holds unrelated data.

What to gather

Have these ready - you will send all four to Comma:

- **Storage account name** - 3-24 chars, lowercase `a-z` and `0-9` only.
- **Container name** - 3-63 chars, lowercase `a-z` , `0-9` , and single (non-consecutive, non-trailing) hyphens. **No dots.**
- **Prefix** - the object-key prefix Comma delivers under, e.g. `team-acme/` (or blank for the container root).
- **Account key** - one of the storage account's two access keys.

Step 1 - a dedicated storage account and container

Create (or confirm) a storage account reserved for Comma delivery, with TLS 1.2, HTTPS-only, and no public blob access - then a private container:

```
# Only if you are creating a new account/container:
az storage account create \
  --name commaarchive01 --resource-group rg-archive --location eastus \
  --sku Standard_LRS --kind StorageV2 \
  --https-only true --min-tls-version TLS1_2 --allow-blob-public-access false

az storage container create \
  --name archive --account-name commaarchive01 --auth-mode login
```

Step 2 - retrieve an account key

```
#!/usr/bin/env bash
set -euo pipefail

# ---- fill these in ----
STORAGE_ACCOUNT="commaarchive01"   # 3-24 lowercase letters/digits
RESOURCE_GROUP="rg-archive"
# -----------------------

echo "Send ONE of these account keys to Comma (store it securely):"
az storage account keys list \
  --resource-group "$RESOURCE_GROUP" --account-name "$STORAGE_ACCOUNT" \
  --query "[0].value" -o tsv
```

Step 3 - verify the key works in Comma's shape

Prove the key can write, read back, and delete under your prefix - the same probe Comma runs on Verify & activate:

```
#!/usr/bin/env bash
set -euo pipefail

STORAGE_ACCOUNT="commaarchive01"; CONTAINER="archive"; PREFIX="team-acme/"
export AZURE_STORAGE_ACCOUNT="$STORAGE_ACCOUNT"
export AZURE_STORAGE_KEY="<the account key>"

KEY="${PREFIX}_comma_probe_$(uuidgen | tr '[:upper:]' '[:lower:]')"
printf 'comma blob export connection probe' > /tmp/comma-probe.txt

az storage blob upload   --container-name "$CONTAINER" --name "$KEY" \
  --file /tmp/comma-probe.txt --overwrite >/dev/null && echo "  write  OK"
az storage blob download --container-name "$CONTAINER" --name "$KEY" \
  --file /tmp/comma-rb.txt >/dev/null \
  && [ "$(cat /tmp/comma-rb.txt)" = "comma blob export connection probe" ] && echo
"  read   OK"
az storage blob delete   --container-name "$CONTAINER" --name "$KEY" >/dev/null &&
echo "  delete OK"

echo "PASS - this account key can deliver in the shape Comma needs."
```

If a step fails, the key lacks that access on the container, or the account/container name is wrong.

Step 4 - send Comma the details

Send your account team, over a secure channel: **provider (Azure Blob Storage), storage account, container, prefix, and one account key.** Comma runs a one-time Verify & activate probe and turns on delivery. Delivery is go-forward from activation.

Optional hardening

- **Restrict by source IP.** All delivery leaves Comma from one static egress IP. Add a storage IP network rule for it, and (once your own admin IP is also allowed) set the account's default action to Deny. Ask your account team for the current egress IP first.

```
az storage account network-rule add \
  --resource-group rg-archive --account-name commaarchive01 --ip-address
<comma-egress-ip>
# then, carefully:
az storage account update \
  --resource-group rg-archive --name commaarchive01 --default-action Deny
```

- **Rotate freely.** Storage accounts have two keys - hand Comma one, and you can rotate the other on your own schedule.

Good to know

- **TLS always.** Comma targets `https://<account>.blob.core.windows.net` on port 443 - never a plaintext or custom endpoint.
- **Completeness proof.** Azure does not server-validate the upload checksum on the block-blob path, so per batch Comma writes a `manifest.json` listing every object with its SHA-256 and byte size - that manifest is your completeness proof (the probe path does verify by reading its object back).
- **Deterministic, idempotent.** A retried batch re-writes the same blobs with byte-identical content, so re-delivery is always safe.

> **ALSO AVAILABLE ONLINE**
>
> The same guide lives at docs.commacompliance.com under Guides - "Archive export to Azure Blob." Your Comma account team provisions the destination once you hand over the details.