

Sino Control Hub on Google Cloud Run

The hub as one Cloud Run service, its database on Cloud SQL reached over a private IP, secrets in Secret Manager

Sino Privacy Vault (<https://sinosecure.app>)

These steps have not yet been checked against a live Google Cloud project. Tell us anything that differs.

1. Image

Push the hub image (`sino-hub:<version>` , from Sino with early access) to Artifact Registry:

```
gcloud artifacts repositories create sino --repository-format=docker --
location=<region>
gcloud auth configure-docker <region>-docker.pkg.dev
docker tag sino-hub:<version> <region>-
docker.pkg.dev/<project>/sino/sino-hub:<version>
docker push <region>-docker.pkg.dev/<project>/sino/sino-hub:<version>
```

2. Postgres

Make a Cloud SQL for PostgreSQL instance with a **private IP** in your VPC, plus a `sino_hub` database and user. The hub connects with plain JDBC, so use the private IP rather than the Cloud SQL connector socket. The address is `jdbc:postgresql://<private ip>:5432/sino_hub?sslmode=require`.

Prefer MySQL? Use Cloud SQL for MySQL instead, with a `jdbc:mysql://<host>:3306/sino_hub?sslMode=verify-ca` address (see the *Deployment Guide*, section 5). Cloud SQL's certificates come from the instance's own CA and do not name its IP, so use `sslMode=verify-ca` with `serverSslCert=` set to that CA. The rest is the same.

3. Secrets

```
openssl rand -base64 32 | gcloud secrets create sino-hub-master-key --
data-file=-
printf '%s' '<sino_hub password>' | gcloud secrets create sino-hub-db-
password --data-file=-
```

Give the service's account `roles/secretmanager.secretAccessor` on both. Back up the master key outside Google Cloud as well: without it the database cannot be opened.

4. The Service

```
gcloud run deploy sino-hub --image <region>-
docker.pkg.dev/<project>/sino/sino-hub:<version> --region <region> \
  --port 8080 --memory 1Gi --cpu 1 \
  --min-instances 1 --max-instances 1 --no-cpu-throttling \
  --network <vpc> --subnet <subnet> --vpc-egress private-ranges-only \
  --set-secrets SINO_HUB_MASTER_KEY=sino-hub-master-
key:latest,SINO_HUB_DATABASE_PASSWORD=sino-hub-db-password:latest \
  --set-env-vars SINO_HUB_DATABASE_URL='jdbc:postgresql://<private
ip>:5432/sino_hub?
sslmode=require',SINO_HUB_DATABASE_USER=sino_hub,SINO_HUB_TRUST_PROXY=fa
lse \
  --allow-unauthenticated
```

- `--no-cpu-throttling` keeps the CPU on between requests. Without it the hub's timers stall: SIEM delivery, recovery-copy mirroring and clean-up.
- `--min-instances 1 --max-instances 1` keeps exactly one hub. During a deploy, Cloud Run briefly runs the old and new revisions side by side. That is harmless for a few seconds, but don't split traffic between revisions.
- `--allow-unauthenticated` is right: devices and the console sign in to the hub themselves. Cloud Run's IAM check would shut them out.
- Cloud Run allows a request 300 seconds by default, longer than the 3 minutes a PC's stream lasts. Keep the request timeout at 300 seconds or more.

- Add a startup probe and a liveness probe on HTTP `/healthz`, port 8080 (`--startup-probe` / `--liveness-probe`, or in the service YAML).
 - **Rate limits.** The hub counts the last address in `X-Forwarded-For` when `SINO_HUB_TRUST_PROXY=true`. It is not yet confirmed that Cloud Run's front end puts the client's address last. Until it is, leave the setting `false`; the limits then count per Google front end, which is coarser. Behind an external Application Load Balancer, add Cloud Armor rate limiting as well.
-

5. Your Domain

Map `hub.example.com` to the service, either with a domain mapping or an external Application Load Balancer with a Google-managed certificate. This address is permanent: every device keeps it. Then read the setup code:

```
gcloud run services logs read sino-hub --region <region> --limit 100 |  
grep "setup code"
```

Open the address, claim the hub with your passkey, and sign its certificate in the Sino app (*Deployment Guide*, section 3, step 5).

6. Storage

Google Cloud Storage is not a storage target yet. Vaults use the company's Azure Blob, AWS S3, Backblaze B2 or Cloudflare R2 storage, which the hub reaches over outbound HTTPS. With `--vpc-egress private-ranges-only` that traffic goes straight out; with `all-traffic`, add a Cloud NAT.