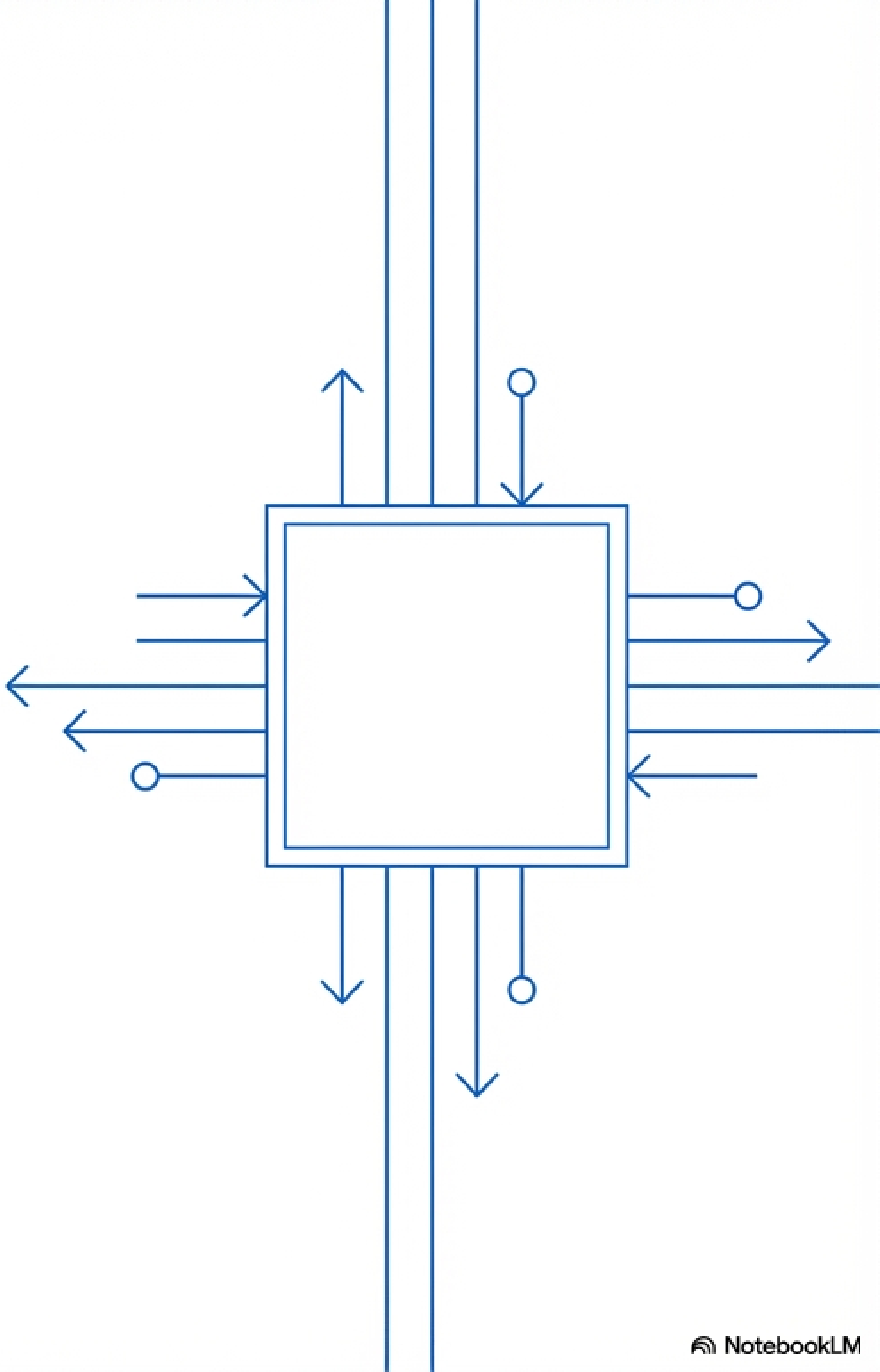
MODULE ANALYSIS // v1.0

CloudRunApp Module Analysis

Architectural Overview, Security Audit & Enhancement Roadmap for the RAD Platform

```
Target Artifact: modules/CloudRunApp
Audit Type:      System Architecture & Security
Platform:        Google Cloud Run v2
```



NotebookLM

The Orchestration Core of the RAD Platform

The Capabilities

Compute:
Deploys Cloud Run v2 Services with 'Startup CPU Boost'.

Data:
Integrates Cloud SQL, NFS, and GCS (including GCS Fuse).

Lifecycle:
Manages initialization jobs (migrations, backups).

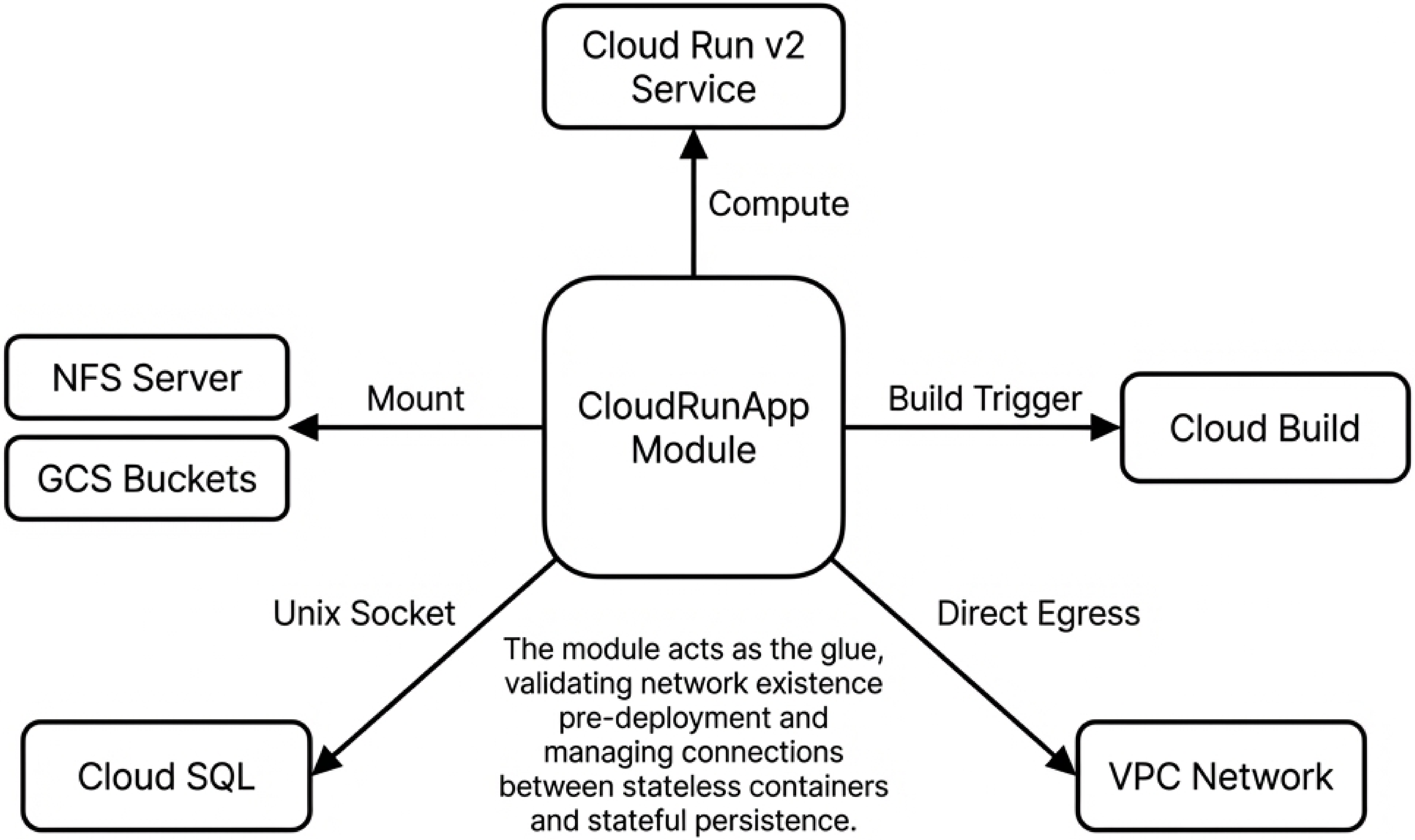
CI/CD:
Built-in support for Cloud Build triggers.

The `modules/CloudRunApp` is the foundational building block for deploying containerized applications on Google Cloud. It orchestrates the entire ecosystem of **networking, storage, databases**, and **observability**.

CRITICAL ATTENTION

While the module utilizes cost-effective Direct VPC Egress, a critical logic gap in IAM configuration currently leaves services publicly accessible by default.

Ecosystem Integration & Architecture



The module acts as the glue, validating network existence pre-deployment and managing connections between stateless containers and stateful persistence.

Identity Architecture: The Application Runtime

Subject: `cloud_run_sa` (Service Account)

secretAccessor

`roles/secretmanager.secretAccessor`

Critical. Allows reading database passwords and environment variables from Secret Manager.

objectAdmin

`roles/storage.objectAdmin`

Grants full control over app-specific storage buckets (e.g., user uploads).

legacyBucketReader

`roles/storage.legacyBucketReader`

Provides metadata access, specifically required for legacy frameworks like Django (django-storages`) compatibility.

Design Philosophy: Least-Privilege operations required for application function.

Identity Architecture: The Builder Persona

Subject: `cloud_build_sa` (Service Account)

run.developer

`roles/run.developer`

Empowers the build process to deploy new revisions to the Cloud Run service.

serviceAccountUser

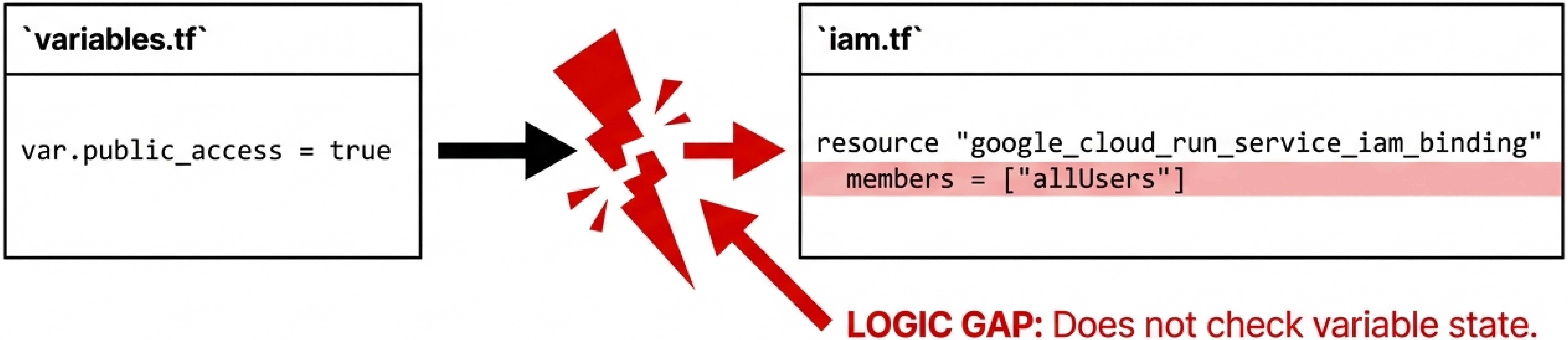
`roles/iam.serviceAccountUser`

Allows Cloud Build to ‘impersonate’ the Application Identity (`cloud_run_sa`) during the deployment process, ensuring the runtime identity is correctly attached the runtime identity is correctly attached to the new revision.

Design Philosophy: Permissions scoped strictly to deployment and revision management.

Audit Finding: Unrestricted Public Access

SEVERITY: HIGH



`variables.tf`

```
var.public_access = true
```

`iam.tf`

```
resource "google_cloud_run_service_iam_binding"
  members = ["allUsers"]
```

LOGIC GAP: Does not check variable state.

The Flaw: The variable `public_access` exists but is ignored in the IAM logic.

The Result: The module grants `roles/run.invoker` to `allUsers` (public internet) by default.

Remediation: Update `iam.tf` logic to wrap the `allUsers` binding in a conditional check against `var.public_access`.

Compute Specifications & Runtime Logic

Environment

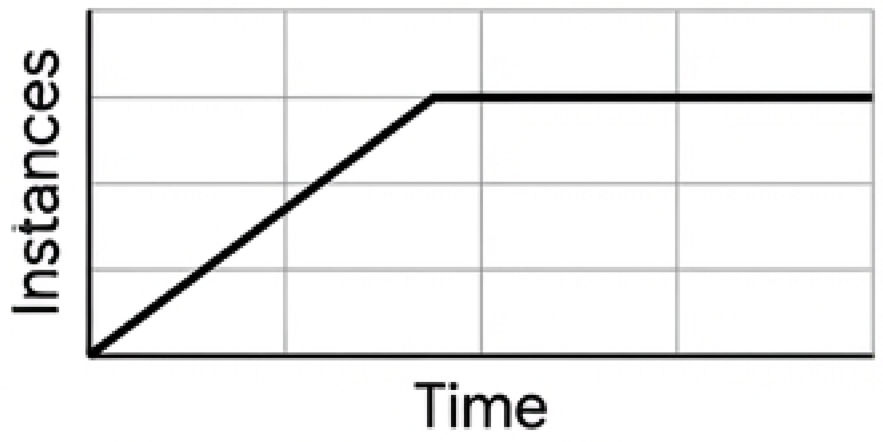
Cloud Run v2
Gen2 Execution Environment

Performance Tuning

Configurable vCPU & Memory Limits

Feature: Startup CPU Boost
(Reduces cold-start latency)

Scaling Strategy

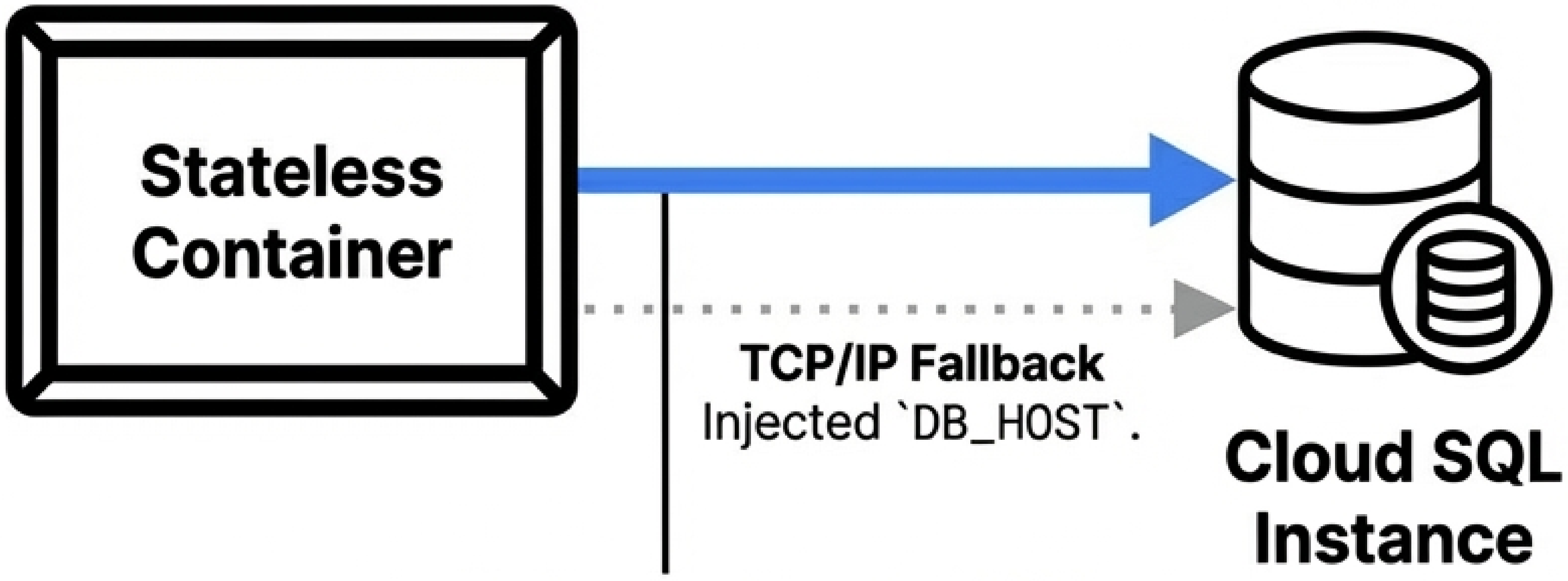


Scale-to-Zero (`min: 0`)
Auto-scales to `max_instance_count`

Protocols

Default: HTTP/1.1
Supported: H2C (HTTP/2 Cleartext) for gRPC / Service Mesh

Data Persistence: Cloud SQL Integration



Unix Socket Volume
Primary. Mounts as `/cloudsql/...`
for low latency & security.

Discovery Model: The module does *not* provision Cloud SQL. It uses `scripts/core/get-sqlserver-info.sh` to dynamically locate instances.

Security: Credentials auto-retrieved via Secret Manager (No hardcoded passwords).

Storage Strategy: NFS vs. Object Storage

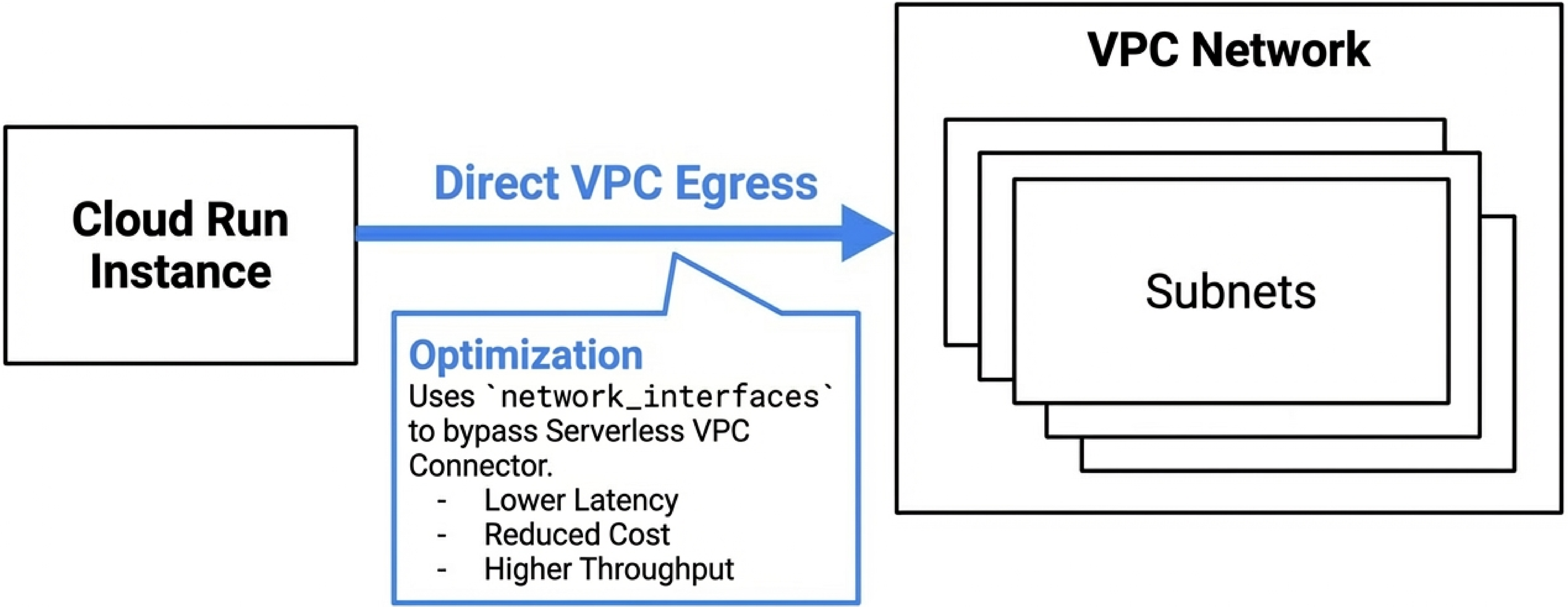
NFS (Network File System)

- **Mechanism**: Auto-detects server via `get-nfsserver-info.sh`.
- **Integration**: Mounts share to `/mnt/nfs`.
- **Use Case**: Shared persistent volume for legacy application state.

GCS (Google Cloud Storage)

- **Mechanism**: Standard Buckets (Versioning/Lifecycle).
- **Integration**: **GCS Fuse**. Mounts buckets as a file system.
- **Use Case**: Transparent file I/O translation to API calls for legacy apps.

Network Topology & Egress Optimization



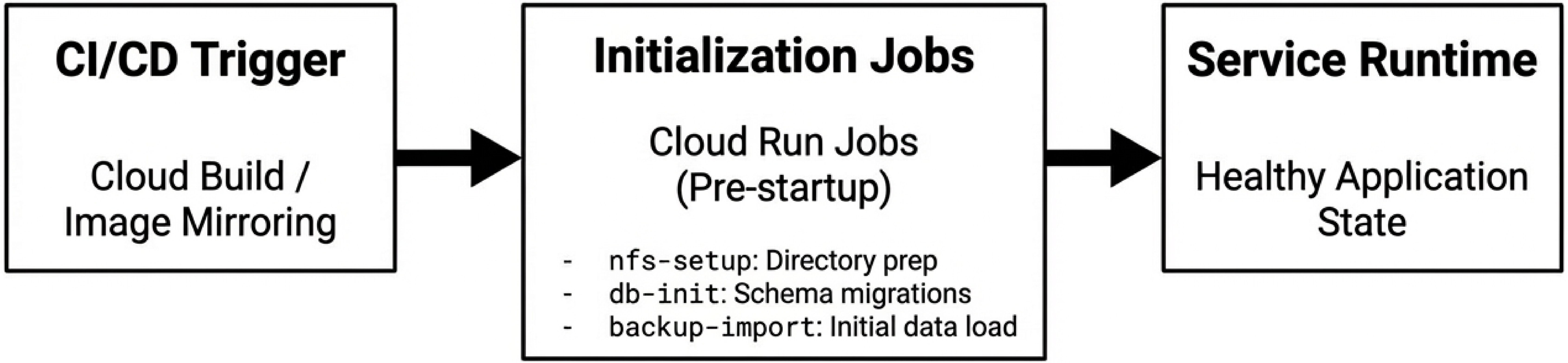
Optimization

Uses `network_interfaces` to bypass Serverless VPC Connector.

- Lower Latency
- Reduced Cost
- Higher Throughput

Includes pre-flight check `scripts/core/check_network.sh` to validate VPC existence.

Operational Lifecycle & CI/CD



CI/CD Trigger

Cloud Build /
Image Mirroring

Initialization Jobs

Cloud Run Jobs
(Pre-startup)

- `nfs-setup`: Directory prep
- `db-init`: Schema migrations
- `backup-import`: Initial data load

Service Runtime

Healthy Application
State

Roadmap: Security & Resilience

Security Upgrades

- **Fix IAM Logic**: Strictly enforce `var.public_access` condition.
- **Cloud Armor**: Integrate WAF/DDoS policies.
- **Granular Invokers**: Implement `allowed_invokers` whitelist for specific emails/groups.

Resilience Upgrades

- **Traffic Splitting**: Enable Canary or Blue/Green deployments (e.g., 10% traffic to new revision).
- **Multi-Region**: Harden active-active global load balancing logic.

Roadmap: Performance & DevEx

Performance Enhancements

- **Native Redis**: Add support for Cloud Memorystore provisioning/discovery (Critical for Django/Magento).
- **Cloud CDN**: Integrate via Load Balancer for static asset caching.

Developer Experience

- **Cloud Buildpacks**: Support source-based deployment.

  Allows deploying directly from source code without maintaining a `Dockerfile`.

Strategic Recommendations

Priority	Action Item	Impact
CRITICAL	Remediate `iam.tf` Public Access Logic	Prevent accidental public exposure.
HIGH	Implement Cloud Armor / WAF	DDoS protection and endpoint security.
MEDIUM	Native Redis (Memorystore) Support	Standardized caching architecture.
LOW	Cloud Buildpacks Integration	Lower developer barrier to entry.

Documentation & Resources

- RAD Console
- GitHub Repository
- Module Path: modules/CloudRunApp
- Discovery Scripts: `scripts/core/`

System Audit Complete.