Azure LightRAG Migration — Complete Runbook

Azure LightRAG Migration — Complete Runbook

> **Domain note (2026-05-17):** This doc was written when hostnames were `lightrag.basicconsulting.no` and `ollama.basicconsulting.no`. Both have since migrated to `lightrag.alai.no` and `ollama.alai.no`. Historical command examples below retain original hostnames for accuracy; use alai.no equivalents in live ops.

Status: COMPLETED 2026-04-18
Team Lead: Kelsey Hightower (FlowForge)
Architect: Petter Graff (CodeCraft)
Data Lead: Martin Kleppmann (CodeCraft)
Validator: Angie Jones (Proveo)
Documentation: Skillforge

Operational Update — 2026-05-20

MC #101607 repaired a query-path regression after Azure LightRAG health was green but `/query` returned HTTP 500 because `LLM_MODEL=qwen3:8b-q8_0` was not available behind the Ollama tunnel. Azure `/home/alai-admin/lightrag/.env` now uses `LLM_MODEL=llama3.1:8b`, and the `lightrag` container was force-recreated without deleting volumes. Direct Azure endpoint `http://20.240.61.67:9621` verified: `/health` healthy and `/query` HTTP 200. Local Anvil/Pi mock config currently uses the Azure direct URL as `lightrag.base_url`.

MC #101611 added client-side Cloudflare Access service-token support for the LightRAG wrapper and key consumers. When `lightrag.base_url=https://lightrag.alai.no`, provide `LIGHTRAG_CF_ACCESS_CLIENT_ID` and `LIGHTRAG_CF_ACCESS_CLIENT_SECRET` (or generic `CF_ACCESS_CLIENT_ID` / `CF_ACCESS_CLIENT_SECRET`). Do not commit secrets. Do not switch canonical `lightrag.base_url` from Azure direct to the public URL until a real service token validates `/health`

and `/query` through Cloudflare Access.

Evidence: `/tmp/verify-101607/SUMMARY.md`, `/tmp/verify-101609/SUMMARY.md`, `/tmp/verify-101611/SUMMARY.md`.

CRITICAL Security Update — 2026-06-18/19

MC #103912: Public exposure incident and migration to app-layer JWT authentication.

Root Cause (2026-06-18)

Boot.sh/discover.js reported "LightRAG DOWN" (false negative). NSG `vm-alai-lightragNSG` only allows port 9621 from `46.46.240.0/20` + Cloudflare ranges; orchestrator host egress IP (`92.221.168.61`) was not whitelisted. LightRAG container was healthy throughout.

Security Incident Timeline

- **18:19 UTC:** New Cloudflare tunnel `lightrag-alai` (c79ffe4d-0be9-40c6-9eaa-608edeb307db) created on Azure VM to resolve access issue. DNS `lightrag.alai.no` updated to point to tunnel.
- **18:19–18:38 UTC (~19 min): PUBLIC EXPOSURE.** Tunnel published `lightrag.alai.no` with NO enforced authentication. Full knowledge graph (15,840 docs incl. internal company/client/partner names) queryable from open internet.
- **18:38 UTC:** Tunnel stopped after external-vantage verification confirmed HTTP 200 leak (builder's "302 BLOCKED" claim was tested from bypass-whitelisted IP and invalid).

Why Cloudflare Access Failed:

Attempted 3 times:

1. Wrong AUD tag (used app ID instead of correct AUD `45433679774e5bb11a3a5c284cf3a71e9f8865c93c513f5cc204e382e96cff8d`)
2. Wrong team name (`alai` instead of `alai-no`)
3. `originRequest.access` enforces user JWT tokens (browser login), NOT service token headers

KEY LESSON: Verify allowlist/auth ONLY from a vantage NOT in any bypass list. CF Access IP-Bypass policy whitelists `92.221.168.61`; testing from that IP always bypassed Access regardless of config.

Final Solution: App-Layer JWT Authentication

Enforcement Point: LightRAG FastAPI application (sbnb/lightrag container), NOT Cloudflare edge.

Configuration:

VM path: `/home/alai-admin/lightrag/.env`

```
AUTH_ACCOUNTS=alai-system:<password_hash>
TOKEN_SECRET=<jwt_secret>
WHITELIST_PATHS=/health
```

Credentials: Vaultwarden item `67dc69b5-b1cb-4892-970e-b4d60380378f` (LightRAG API Auth).

Auth Flow:

1. Client POSTs to `/login` with username + password (form data)
2. Server validates credentials, returns JWT (48h expiry)
3. Client includes `Authorization: Bearer <token>` on subsequent requests
4. `/health` is PUBLIC (whitelisted for boot.sh probes)
5. `/query`, `/insert`, `/api/*` require valid JWT (HTTP 401 without auth)

Verification (external-vantage required):

```
# Unauthenticated MUST return 401
curl -i -X POST https://lightrag.alai.no/query \
  -H 'Content-Type: application/json' -d '{"query":"test"}'
# Expected: HTTP/1.1 401 Unauthorized

# Health endpoint PUBLIC
curl -s https://lightrag.alai.no/health | jq -r '.status'
# Expected: "healthy"

# Authenticated access
TOKEN=$(curl -s -X POST 'https://lightrag.alai.no/login' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -d 'username=alai-system&password=<from_vaultwarden>' \
  | jq -r '.access_token')

curl -s -X POST https://lightrag.alai.no/query \
  -H 'Content-Type: application/json' \
  -H "Authorization: Bearer $TOKEN" \
```

```
  -d '{"query":"test","mode":"naive","top_k":1}' | jq -r '.response'
# Expected: HTTP 200 + query results
```

Architecture:

```
External Client (unauthenticated)
  ↓ HTTPS POST /query
Cloudflare (lightrag.alai.no)
  ↓ Tunnel (c79ffe4d)
Azure VM cloudflared
  ↓ HTTP localhost:9621
LightRAG FastAPI Server
  → JWT Validator Middleware
  → HTTP 401 Unauthorized  BLOCKED

External Client (authenticated with JWT)
  ↓ HTTPS POST /query + Authorization: Bearer <jwt>
Cloudflare → Tunnel → VM cloudflared → localhost:9621
LightRAG FastAPI Server
  → JWT validates (TOKEN_SECRET, 48h expiry)
  → HTTP 200 + query results  ALLOWED
```

Known Residuals

- Vestigial CF Access Application (c62b46b1-43f4-4967-9b99-cabfefb6b99b) + IP-Bypass policy still exist but not enforced (app-layer takes precedence).
- `/health` endpoint exposes doc counts + config (ollama host, storage backends) publicly. Accepted for health checks.
- The AUTH_ACCOUNTS password appeared in plaintext in agent reports/evidence during initial fix; recommend rotation as hygiene.

Rotating LightRAG Credentials

```
# Generate new password hash
NEW_PASS=$(openssl rand -hex 32)
HASH=$(echo -n "$NEW_PASS" | sha256sum | awk '{print $1}')

# Update on VM
ssh -i ~/.ssh/azure_alai alai-admin@20.240.61.67
cd ~/lightrag
```

```
# Edit .env: AUTH_ACCOUNTS=alai-system:<new_hash>
docker compose restart lightrag

# Test
curl -s -X POST 'https://lightrag.alai.no/login' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -d "username=alai-system&password=$NEW_PASS" | jq -r '.access_token'

# Update Vaultwarden item 67dc69b5-b1cb-4892-970e-b4d60380378f with new password
```

boot.sh / discover.js Integration

Config: `~/system/tools/alai-config-mock.json`

```
{
  "lightrag.base_url": "https://lightrag.alai.no"
}
```

Credentials loaded from environment (NOT committed to git):

```
export LIGHTRAG_USERNAME="alai-system"
export LIGHTRAG_PASSWORD="<from_vaultwarden>"
```

boot.sh/discover.js authenticate on first request, cache JWT for 48h, refresh when expired.

Evidence: `/tmp/evidence-103912/app-layer-auth-proof.md`, `/tmp/evidence-103912/verification-final.md`

Executive Summary

Why migrated: Docker Desktop failed 3 times on 2026-04-18, causing LightRAG outages impacting all ALAI knowledge operations (discover.js, autocoder.js, retrieval-orchestrator.js). Local dependency became unacceptable single point of failure.

What changed:

- LightRAG + Neo4j moved from Mac Studio Docker Desktop → Azure VM `vm-alai-lightrag` (swedencentral)
- Ollama remains on Mac Studio but exposed via Cloudflare tunnel with Zero Trust IP whitelist

- 8 consumer files updated to use `https://lightrag.basicconsulting.no` instead of `http://localhost:9621`
- Data migrated: 497MB snapshot from Azure Blob backup 20260418-085317 (zero data loss)

Result:

- ✅ Docker Desktop crashes no longer affect LightRAG availability
- ✅ Query latency acceptable: p50 ~2-3s vs ~1s local (30-60ms tunnel overhead + network)
- ✅ Rollback capability preserved: 5-15 min return to local if needed
- ✅ Azure cost: ~$30/month (B2s_v2 VM), pulled from credits

Architecture Diagram

```
sequenceDiagram
    participant Consumer as Mac Studio Consumer<br/>(discover.js, autocoder.js, etc.)
    participant CF1 as Cloudflare<br/>lightrag.basicconsulting.no
    participant Tunnel1 as Mac Studio<br/>cloudflared tunnel
    participant VM as Azure VM<br/>20.240.61.67:9621<br/>LightRAG container
    participant CF2 as Cloudflare<br/>ollama.basicconsulting.no
    participant Tunnel2 as Mac Studio<br/>cloudflared tunnel
    participant Ollama as FORGE<br/>10.0.0.2:11434<br/>Ollama service

    Consumer->>CF1: HTTPS query request
    CF1->>Tunnel1: Route via tunnel
    Tunnel1->>VM: Forward to Azure VM:9621
    VM->>CF2: HTTPS request for LLM inference
    CF2->>Tunnel2: Route via tunnel (Zero Trust IP check)
    Tunnel2->>Ollama: Forward to FORGE:11434
    Ollama-->>Tunnel2: Inference response
    Tunnel2-->>CF2: Return
    CF2-->>VM: LLM result
    VM->>VM: Build knowledge graph (Neo4j)
    VM-->>Tunnel1: Query response
    Tunnel1-->>CF1: Return
    CF1-->>Consumer: HTTPS response
```

Key insight: Cloud LightRAG talks back to on-prem Ollama via second tunnel. Both services behind Cloudflare Zero Trust.

Resources Created

Azure Resource Group

- **Name:** `rg-alai-lightrag`
- **Location:** swedencentral
- **Purpose:** Dedicated to LightRAG stack

Virtual Machine

- **Name:** `vm-alai-lightrag`
- **Size:** Standard_B2s_v2 (2 vCPU, 8 GB RAM, Intel x86_64)
- **OS:** Ubuntu 22.04 LTS
- **Public IP:** 20.240.61.67 (static)
- **OS Disk:** 30GB Premium SSD
- **Data Disk:** Not used (Docker volumes on OS disk sufficient for now)
- **SSH:** `ssh -i ~/.ssh/azure_alai alai-admin@20.240.61.67`
- **Docker:** 29.4.0
- **Docker Compose:** 2.30.3

Network Security Group (NSG)

Name: `vm-alai-lightragNSG`

Rule Name	Priority	Direction	Port	Source	Purpose
`default-allow-ssh`	1000	Inbound	22	46.46.251.40/32	SSH from Mac Studio ISP
`allow-lightrag-macstudio`	100	Inbound	9621	46.46.251.40/32	Direct access (backup)
`allow-cloudflare-lightrag`	110	Inbound	9621	Cloudflare IP ranges	Tunnel ingress

Important: Mac Studio ISP IP (46.46.251.40) is residential and may rotate. When rotation happens, SSH and direct API access will fail. Update NSG rules accordingly (see Troubleshooting).

Cloudflare DNS Records

- **lightrag.basicconsulting.no** → CNAME to `3315a609-7934-45c5-ad0c-56d86d16374d.cfargotunnel.com`

- Tunnel config: `service: http://20.240.61.67:9621`
- Routes external consumers to Azure VM via Mac Studio tunnel (relay)

- **ollama.basicconsulting.no** → CNAME to `3315a609-7934-45c5-ad0c-56d86d16374d.cfargotunnel.com`
  - Tunnel config: `service: http://10.0.0.2:11434`
  - Routes Azure VM back to FORGE Ollama via Mac Studio tunnel
  - Zero Trust policy: IP whitelist includes Azure VM egress IP + Mac Studio

---

Data Migration

Source

- **Azure Blob Storage:** `plockfrontstaging/lightrag-backup/20260418-085317/`
- **Size:** 497MB compressed (4 tarballs + manifest + README)
- **Volumes:**
  - `lightrag-data.tar.gz` (312MB) — KV store + inputs
  - `lightrag-kg.tar.gz` — Knowledge graph files
  - `lightrag-cache.tar.gz` — LLM response cache
  - `lightrag-neo4j-data.tar.gz` (169MB) — Neo4j graph entities + relations

Restore Process

1. Download from Azure Blob to VM `/tmp/restore/`
2. `shasum -a 256 -c MANIFEST.sha256` — verified all 4 tarballs
3. Created 4 Docker volumes
4. Extracted each tarball into its volume using Alpine container
5. Started LightRAG + Neo4j containers
6. Verified entity count in Neo4j matched pre-migration snapshot

Data loss: ZERO. Snapshot taken immediately before migration.

Consumer Files Cut Over

8 files updated from `http://localhost:9621` → `https://lightrag.basicconsulting.no`:

File	Purpose
`~/system/tools/discover.js`	Universal search (tools, agents, docs, RAG)
`~/system/tools/lightrag.js`	LightRAG client wrapper

File	Purpose
`~/system/tools/autocoder.js`	Code generation with RAG context
`~/system/tools/lightrag-bulk-upload.js`	Batch document ingestion
`~/system/tools/lightrag-migrate.js`	Migration utility
`~/system/tools/lightrag-outbox-ingest.js`	Outbox processor
`~/system/tools/retrieval-orchestrator.js`	Multi-source retrieval coordinator
`~/system/tools/system-regression.sh`	Health check suite

Pre-cutover backups: Not created (files tracked in Git, easy revert via `git restore`).

Operational Procedures

Daily Health Check

```
# From Mac Studio
curl https://lightrag.basicconsulting.no/health

# Expected response:
# {"status":"healthy","working_directory":"/app/data", ...}
```

SSH to VM

```
ssh -i ~/.ssh/azure_alai alai-admin@20.240.61.67
```

Docker Container Management

```
# On Azure VM
cd ~/lightrag
docker compose ps              # Check status
docker compose logs -f         # Tail logs
docker compose restart         # Restart services
docker compose down && docker compose up -d  # Full restart
```

Health Check from VM (tests Ollama tunnel)

```
# On Azure VM
curl -s https://ollama.basicconsulting.no/api/tags | jq '.models | length'
# Should return model count (e.g., 12)
```

Azure Cost Check

```
# From Mac Studio
az consumption usage list --start-date $(date -u -v-7d +%Y-%m-%d) --end-date $(date -u +%Y-%m-%d) -o table
```

Stop LightRAG (for maintenance)

```
# On Azure VM
cd ~/lightrag
docker compose stop
# Restart after maintenance
docker compose start
```

Rollback Procedure — CRITICAL

When to rollback:

- Azure VM becomes unstable or unresponsive for >30 min
- Tunnel failures persist despite troubleshooting
- Cost overrun detected
- Any catastrophic issue requiring immediate local restore

Expected rollback time: 5-15 minutes
Data loss risk: ZERO (local volumes preserved 7 days post-cutover)

Step 1: Revert Consumer URLs

```
# On Mac Studio
cd ~/system/tools
for file in discover.js lightrag.js autocoder.js lightrag-bulk-upload.js \
            lightrag-migrate.js lightrag-outbox-ingest.js \
            retrieval-orchestrator.js system-regression.sh; do
```

```
  sed -i '' 's|https://lightrag.basicconsulting.no|http://localhost:9621|g' "$file"
done

# Verify
grep -l "localhost:9621" *.js *.sh
# Should list all 8 files
```

Step 2: Restart Local Docker LightRAG

```
cd ~/system/docker/lightrag
docker compose up -d

# Wait for healthy status (30-60s)
docker compose ps
```

Step 3: Verify Local Service

```
curl http://localhost:9621/health
# Expected: {"status":"healthy", ...}

# Run regression suite
bash ~/system/tools/system-regression.sh
# LightRAG checks should PASS
```

Step 4: Deprovision Azure VM (optional, when convenient)

```
az group delete --name rg-alai-lightrag --yes --no-wait
# Deletes VM, NSG, disks, public IP
# Cloudflare DNS records remain (harmless)
```

Post-rollback actions:

1. Update `~/system/docs/runbooks/lightrag-backup.md` to revert to local backup flow
2. Notify Alem in Slack #ops-alai
3. Create MC task for post-mortem

Troubleshooting

Issue: "model not found" errors in LightRAG logs

Cause: Cloudflare tunnel `ollama.basicconsulting.no` routing to wrong backend.

Diagnosis:

```
# On Azure VM
curl -s https://ollama.basicconsulting.no/api/tags | jq '.models[].name'
# Should list qwen2.5-coder:32b-instruct-q8_0, bge-m3:latest, etc.
```

Fix:

1. Check Mac Studio tunnel config: `cat ~/.cloudflared/config.yml | grep -A2 ollama`
2. Should be `service: http://10.0.0.2:11434` (FORGE), NOT `http://localhost:11434` (ANVIL)
3. If wrong: edit config, restart tunnel: `launchctl kickstart -k gui/$(id -u)/com.john.cloudflared`

Related incident: 2026-04-18 mid-migration — tunnel initially pointed to ANVIL, causing q8_0 model not found. Changed to FORGE, resolved immediately.

Issue: SSH connection timeout

Cause: Mac Studio ISP rotated IP; NSG rule `default-allow-ssh` still has old IP (46.46.251.40/32).

Diagnosis:

```
# On Mac Studio
curl https://ifconfig.co
# Compare to NSG rule source IP
az network nsg rule show -g rg-alai-lightrag --nsg-name vm-alai-lightragNSG -n default-allow-ssh --query "sourceAddressPrefix"
```

Fix:

```
NEW_IP=$(curl -s https://ifconfig.co)
az network nsg rule update \
  -g rg-alai-lightrag \
  --nsg-name vm-alai-lightragNSG \
```

```
  -n default-allow-ssh \
  --source-address-prefixes "${NEW_IP}/32"

# Verify
ssh -i ~/.ssh/azure_alai alai-admin@20.240.61.67
```

Note: Use `ifconfig.co` or `icanhazip.com`, NOT `ifconfig.me` (returns CDN IP on some networks).

Issue: "connection refused" from consumer scripts

Cause: Cloudflare tunnel down or misconfigured.

Diagnosis:

```
# From Mac Studio
curl https://lightrag.basicconsulting.no/health
# If timeout/connection refused, tunnel is down

# Check tunnel process
ps aux | grep cloudflared
# Should show cloudflared running with config ~/.cloudflared/config.yml

# Check tunnel logs
tail -f ~/Library/Logs/cloudflared/cloudflared.log
```

Fix:

```
# Restart tunnel
launchctl kickstart -k gui/$(id -u)/com.john.cloudflared

# Wait 10-15s, retry
curl https://lightrag.basicconsulting.no/health
```

Issue: Slow query responses (>10s p95)

Cause 1: Network path issue (Mac Studio → Cloudflare → Azure → Cloudflare → Mac Studio).

Diagnosis:

```
# On Azure VM
time curl -s https://ollama.basicconsulting.no/api/tags > /dev/null
# Should be <100ms

# From Mac Studio
time curl -s https://lightrag.basicconsulting.no/health > /dev/null
# Should be <200ms
```

Cause 2: Ollama FORGE overloaded (other tasks using models).

Diagnosis:

```
# On Mac Studio
curl http://10.0.0.2:11434/api/ps
# Check running models
```

Fix: Identify and throttle/stop competing workloads on FORGE.

Issue: Neo4j "unable to allocate memory"

Cause: B2s_v2 has 8GB RAM; Neo4j + LightRAG + OS overhead can approach limit.

Diagnosis:

```
# On Azure VM
docker stats --no-stream
# Check memory usage percentages

free -h
# Check system memory
```

Fix (short-term):

```
# Restart containers to clear caches
cd ~/lightrag
docker compose restart
```

Fix (long-term): Upgrade to Standard_B2s_v2 → Standard_B4ms (4 vCPU, 16GB RAM, ~$60/month).

Cross-References

- **Ollama Tunnel:** ollama-cloudflare-tunnel.md — tunnel config, Zero Trust policy, failure modes
- **Backup Flow:** lightrag-azure-backup.md — updated for cloud-resident LightRAG
- **Migration Plan:** `~/system/specs/lightrag-azure-migration-plan.md` — full task breakdown (14 tasks, 5 phases)

Validation Evidence

Completed by: Angie Jones (Proveo), 2026-04-18

- ☐ Full round-trip: `discover.js "lightrag"` returns hits from cloud graph
- ☐ Ingest test: new document ingested via `lightrag-outbox-ingest.js`, confirmed in Neo4j
- ☐ Docker Desktop killed for 10 min: all ALAI functions remained operational
- ☐ Latency test: 20 queries — p50=2.3s, p95=7.1s (acceptable vs local p50=1.1s, p95=3.2s)
- ☐ Azure cost projection: 24h run = $1.20 (prorated to ~$36/month; budget set at $30/month, within tolerance)
- ☐ First Azure-native backup: 20260418-203045 snapshot uploaded to Blob (512MB)

Evidence bundle: `~/system/evidence/azure-lightrag-migration-20260418/SUMMARY.md`

Next Steps

1. **Monitor for 7 days** — track latency, cost, uptime. If stable, delete local Docker volumes.
2. **Update backup flow** — migrate launchd plist to SSH-based Azure VM snapshot (see lightrag-azure-backup.md).
3. **Consider FORGE failover** — expose FORGE Ollama via second tunnel endpoint for redundancy.
4. **Auto-scaling evaluation** — if query volume grows, consider Azure Container Instances or AKS migration.

Document Owner: Skillforge
Last Updated: 2026-04-18
Approved By: Petter Graff (Architecture), Kelsey Hightower (Infra), Alem Basic (CEO)

Revision #6
Created 2026-04-18 23:16:17 UTC by John
Updated 2026-06-18 22:38:24 UTC by John