Bilko Demo Event-Loop Freeze — Root Cause, Fix & Hardening (2026-06-08, MC #103134)

Bilko Demo Event-Loop Freeze — Root Cause, Fix & Hardening (2026-06-08, MC #103134)

1. Incident Summary

Date: 2026-06-08
Service: bilko-api-demo (Cloud Run, europe-north1, tribal-sign-487920-k0)
Live revision at incident: bilko-api-demo-00139-b26 (git sha c8dfb6c)
MC: #103134 (child of #103132)
Detected by: CEO (Alem Basic) — no automated alerts existed

Symptom: POST /api/v1/auth/login and GET /api/v1/health intermittently returned HTTP 504 with server-side latency of exactly 59.999 s (Cloud Run hard timeout). Observed failure rate: 17–25% of requests. Both endpoints froze simultaneously on both Cloud Run instances. A manual container restart did not fix the issue.

Why restart did not help: The blocking code paths are baked into the deployed image (c8dfb6c). The freeze re-triggers the moment any user navigates to Reports or Billing/Usage, which re-executes the same bare JDBC calls on the Netty event-loop thread.

---

2. Root Cause — Causal Chain

Source: 5-agent synthesis (petter-synthesis.md) + source-verified app-layer audit (dev-applayer.md) + data/storage audit (devops2-data.md).

2.1 Ktor Netty Thread Pool Exhaustion (Primary)

Ktor runs on Netty. On a 1-vCPU Cloud Run container with no callGroupSize set (main branch pre-fix), Netty defaults to approximately 2 call-handling threads. With containerConcurrency=8, Cloud Run routes up to 8 simultaneous requests into the container, all competing for those 2 threads.

Any route that executes a bare orgTransaction() call not wrapped in dbQuery{} runs its JDBC work directly on the Netty call thread, blocking it for the duration of the DB round-trip (10–100 ms typical, up to 30 s if HikariCP must wait for a pool slot). With only 2 call threads and 8 concurrent slots, two simultaneous requests to a bare-orgTransaction route exhaust both call threads. The event loop is frozen.

Confirmed bare orgTransaction sites on main at incident time:

File	Line(s)	Route
ReportRoutes.kt	26	GET /reports (country lookup)
ReportRoutes.kt	240	GET /reports/kpo (country pre-check)
ReportRoutes.kt	264	GET /reports/kpo/export/pdf (country pre-check)
ReportRoutes.kt	285	GET /reports/kpo/export/xlsx (country pre-check)
BillingRoutes.kt	214	GET /billing/usage (plan tier + invoice count)
MarketRoutes.kt	2 sites	resolveMarketPlugin call sites
ResourceAccessFilter.kt	multiple	Per-request security filter
Authentication.kt	impersonation path	Per impersonation request

2.2 PermissionService.loadFromDb — New RBAC Blocking Call (Critical, Entra cutover)

The Entra External ID cutover (MC #103140) introduced RBAC via requirePermission(), which is called on every authenticated request across all 17 route files. On first call per role (4 roles: owner/admin/accountant/viewer), resolve() calls loadFromDb():

```
// BEFORE fix — blocking on Netty event-loop thread
private fun loadFromDb(role: String): Set<String> {
    return transaction {  // bare JDBC on Netty thread
        RolePermissionsTable.selectAll()...
    }
}
```

This fires on every cold-start / scale-out event for every authenticated route. Without the fix, the Entra cutover would have immediately re-introduced a freeze worse than the original: 4 role cache-misses per instance restart, each running blocking JDBC on the event loop. Source: /tmp/evidence-103134/rebase-on-entra.md.

2.3 HikariCP connectionTimeout=30,000 ms (Secondary Amplifier)

When the Dispatchers.IO pool workers (correctly dispatched DB calls) hit HikariCP pool pressure — 10 connections shared across 8 concurrent slots — a pool-borrow wait can last up to 30 s per request. Two stacked pool-borrow waits = 60 s = Cloud Run hard kill. This amplifier remains even after the bare-orgTransaction calls are fixed; it is addressed by reducing connectionTimeout to 5 s (P0.3).

2.4 Why /health Freezes (Not a DB Problem)

HealthRoutes.kt lines 10–21 are confirmed DB-independent (zero blocking calls, pure in-memory response). The health freeze is not because health touches the DB. It freezes because both Netty call threads are occupied by blocked orgTransaction callers. Health cannot be scheduled. The TCP startup probe continues to pass because the JVM still has port 4001 bound; only HTTP scheduling is stalled.

2.5 Why Login Freezes (Login Is the Victim, Not the Cause)

AuthRoutes.kt line 150 correctly wraps login in dbQuery{}. Login itself is not a blocking caller. Login freezes because it cannot obtain a call thread to execute — both threads are consumed by the bare-orgTransaction routes.

2.6 Tertiary — Absent Server-Side DB Timeouts

No statement_timeout, idle_in_transaction_session_timeout, or lock_timeout on Cloud SQL (confirmed: databaseFlags null). A single slow query holds a pool connection indefinitely, compounding pool pressure.

2.7 gcsfuse Is Not a Cause of Auth/Health Freezes

gcsfuse GC timestamps do not align with freeze onsets (GC at 06:35:04 UTC, freeze at 06:34:52 and 06:35:43). Login does no file I/O. gcsfuse FUSE blocking I/O affects receipt endpoints only; it is

a separate hygiene issue and was also offloaded to Dispatchers.IO in the fix.

---

3. Fix Applied (MC #103134)

Branch: fix/demo-freeze-on-entra-103134, commit 5252182, rebased on origin/main @ b21b366 (Entra cutover). Merged as PR #279 to main 839ee7f. Source: /tmp/evidence-103134/rebase-on-entra.md.

Code Fixes

File	Change
routes/ReportRoutes.kt (4 sites)	Bare orgTransaction wrapped in dbQuery{}
routes/BillingRoutes.kt (line 214)	Bare orgTransaction in /billing/usage wrapped in dbQuery{}
routes/MarketRoutes.kt (2 sites)	resolveMarketPlugin call sites wrapped in dbQuery{}
plugins/ResourceAccessFilter.kt	fetchResourceOwner + logSecurityViolation made suspend + withContext(IO); Thread.sleep replaced with delay
plugins/Authentication.kt	Impersonation transaction wrapped in withContext(Dispatchers.IO)
services/PermissionService.kt (new finding)	loadFromDb() now uses runBlocking { withContext(Dispatchers.IO) { ... } } — RBAC per-request DB call offloaded from event loop
services/ReceiptService.kt	readLocalDocument + persistLocalIfEnabled made suspend + withContext(IO)
routes/ExpenseRoutes.kt	readLocalDocument call moved out of dbQuery{} (now suspend-safe directly)
plugins/Database.kt	connectionTimeout 30,000 ms reduced to 5,000 ms
routes/HealthRoutes.kt	Added GET /api/v1/health/deep — DB-touching endpoint with withTimeout 3 s + Dispatchers.IO + SELECT 1

Infrastructure Fixes (cloudbuild-stage.yaml + cloudbuild.yaml)

Change	Before	After
Memory	512 Mi	1 Gi

Change	Before	After
Liveness probe	None	httpGet /api/v1/health/deep, port 4001, initialDelaySeconds 30, periodSeconds 30, failureThreshold 2, timeoutSeconds 8
Startup probe type	TCP port 4001, period 240s, failureThreshold 1, timeout 240s	httpGet /api/v1/health, port 4001, period 10s, failureThreshold 3, timeout 5s
containerConcurrency (cloudbuild templates)	80 (default)	8

Build and Test Results

```
./gradlew build -x detekt: BUILD SUCCESSFUL in 12s
Tests: 2561 passed, 1 skipped (EntraLiveCiamTokenWp5Test — assumeTrue on stale token file), 0 failures
```

---

4. Industry-Standard Scorecard

Condensed from petter-synthesis.md section 2. Full findings in layer audit files.

Application Layer Deviations

Component	Status	Finding
Ktor/Netty callGroupSize	DEVIATES (pre-fix)	Defaults to ~2 call threads on 1 vCPU when unset. callGroupSize=32 added in fix; must be confirmed in Cloud Run env (KTOR_CALL_GROUP_SIZE=32 or production profile).
dbQuery{} discipline	FIXED	5 bare orgTransaction sites + PermissionService.loadFromDb all wrapped. Remaining raw transaction{} calls are safely nested inside outer dbQuery{} or on non-hot paths.
Login path	STANDARD	AuthRoutes.kt:150 correctly wraps in dbQuery{}. Login was the victim of thread starvation, not a cause.

Component	Status	Finding
BCrypt dispatcher	DEVIATES (minor)	BCrypt (CPU-bound 250–400 ms) runs on Dispatchers.IO. Should be Dispatchers.Default. Not the freeze cause. Follow-on P1.3.
HikariCP connectionTimeout	FIXED	30,000 ms reduced to 5,000 ms. Pool-borrow wait now fails fast with 503.
Login transaction count	DEVIATES	AuthService.login() makes 3 separate orgTransaction{} calls (3 pool borrows). Follow-on P1.1: consolidate to 1.
Ktor RequestTimeout plugin	DEVIATES	Not installed. Stuck handlers held until Cloud Run kills at 60 s. Follow-on P1.5: add 10 s server-side timeout.
/health/deep endpoint	ADDED	DB-touching SELECT 1, 3 s internal timeout, returns 200 {db:up} or 503.
gcsfuse (ReceiptService)	FIXED (hygiene)	Blocking POSIX FUSE I/O offloaded to withContext(IO). Not the auth/health freeze cause, but blocked safe concurrency > 1–8.

Infrastructure Layer Deviations

Component	Status	Finding
containerConcurrency	DEVIATES	8 with blocking paths present. Industry standard: 1 until all blocking DB/IO paths are on Dispatchers.IO. Reduced from 80 to 8 in fix; full revert to 1 deferred pending gcsfuse SDK swap (P1.2).
minScale/maxScale	DEVIATES	min=2 max=2 — no scaling relief valve. With both instances holding frozen slots, no new instance absorbs traffic. Follow-on P2.2: maxScale=5 + pool=2.
HikariCP pool math	FRAGILE	pool=10, maxScale=2 = 20 connections vs ~22 usable on db-f1-micro. Within bounds by 2 connections only.
Cloud SQL databaseFlags	DEVIATES	Zero flags. No statement_timeout (follow-on MC #103133, blocked on MC #103149), no idle_in_transaction_session_timeout, no lock_timeout.

Component	Status	Finding
Liveness probe	ADDED	HTTP liveness on /health/deep. Without it, frozen containers ran for the entire incident duration with no auto-recovery.
Startup probe	FIXED	TCP 240s/1 threshold → HTTP GET /health 10s/3 threshold.
Cloud Monitoring alerts	DEVIATES	0 policies (gcloud monitoring policies list = 0 items confirmed 2026-06-08T08:54 UTC). Incident was CEO-detected. Follow-on P2.1: 5 alert policies.
Memory	FIXED	512 Mi to 1 Gi. JVM + HikariCP + gcsfuse daemon on 512 Mi was tight.
Cloud SQL public IP	DEVIATES	ipv4Enabled=true. No open authorized networks found (mitigated), but public surface exists. Follow-on: private IP + VPC.

---

5. Deploy Path and Cross-Session Coordination

Source: /tmp/alai/p2p-pairing-evidence/john-7fedd67f-freeze-fix-coordination-20260608.md

This fix was developed in session 7fedd67f, rebased on the Entra External ID cutover branch (b21b366), and coordinated with the Entra/RBAC workstream (MC #103080, Phase 4). The agreed sequence:

1. **Step 1 (complete):** PR #279 merged fix/demo-freeze-on-entra-103134 to main (839ee7f). Stage auto-deploy triggered. Stage revision bilko-api-stage-00531-cub deployed. Proveo PASS issued.
2. **Step 2 (pending):** Demo and production semver tag is owned by the Entra/RBAC workstream (MC #103080). That single semver tag vX.Y.Z ships Entra + RBAC + freeze fix together via bilko-main-deploy trigger. Neither session tags unilaterally.

Deploy pipeline mechanics (tool-verified):

- bilko-stage-auto-deploy: push to branch matching ^main$ triggers cloudbuild-stage.yaml
- bilko-main-deploy: semver tag matching ^v.*$ triggers cloudbuild.yaml (demo, 8 gates + approval required)
- cloudbuild-demo-api.yaml: DEAD CODE — orphaned trigger deleted 2026-05-21. Do not reference or invoke.

6. Proveo Stage Validation — PASS

Source: /tmp/evidence-103134/proveo-stage-verdict.json + /tmp/evidence-103134/proveo-stage-validation.md
Timestamp: 2026-06-08T09:54:10Z | Revision: bilko-api-stage-00531-cub (git sha 839ee7f)

Check	Result	Detail
/health x30 sequential	PASS	0/30 failures, max latency 170 ms
/health x20 concurrent	PASS	0/20 failures, max latency 279 ms
/health/deep x15 (DB-touching)	PASS	Endpoint present, db:up, all under 200 ms
Concurrency starvation test (6 waves x20 = 120 requests)	PASS	0 non-200, 0x 504@60s. Wave 1: 2/20 requests at ~10.8 s HTTP 200 (pool warm-up artifact, not freeze). Waves 2–6: zero over 2 s. Original 504@59.999s pattern NOT observed.
Auth path	PASS	POST /auth/login 410 (legacy retired); POST /auth/entra/session (invalid) 400; GET /invoices (no auth) 401; GET /contacts (no auth) 401. Entra + RBAC enforced.
Probe config	PASS	livenessProbe httpGet /health/deep present; startupProbe httpGet /health present; memory 1Gi; git sha 839ee7f confirmed.

Overall verdict: PASS. The event-loop freeze fix is confirmed on stage. The original 504@59.999s symptom is not reproduced under sustained concurrent load. All DB-touching paths return fast. Entra-only auth path correctly enforced.

---

7. Follow-Up Tasks

MC / Item	Description	Status
MC #103133	Cloud SQL statement_timeout + idle_in_transaction_session_timeout + lock_timeout	Blocked on GCP-write channel MC #103149
MC #103148	Gate wc bug	Open
MC #103173	Claim-gate stale-transcript loop	Open

MC / Item	Description	Status
Task #7	PermissionService pre-warm cache at JVM startup (4-role cache hit on first request, no cold DB miss)	Deferred — recommended polish
P1.1	Consolidate AuthService.login() 3 transactions into 1 (reduce pool borrows per login from 3 to 1)	Deferred
P1.2	Replace gcsfuse with GCS Storage SDK in ReceiptService (prerequisite for safe containerConcurrency > 8)	Deferred
P1.3	Move BCrypt from Dispatchers.IO to Dispatchers.Default (CPU-bound work on correct dispatcher)	Deferred
P1.5	Install Ktor RequestTimeout plugin (force-kill stuck handlers at 10 s, return 503 not 504)	Deferred
P2.1	Create 5 Cloud Monitoring alert policies (5xx rate, p99 latency, instance ceiling, Cloud SQL connections, /health/deep uptime)	Deferred — no blocking dependency
P2.2	Scaling fix: containerConcurrency=1, maxScale=5, pool=2 (requires P1.2 first)	Deferred pending P1.2
P2.6	Add k6 event-loop-freeze regression gate to CI (spec in test-observability.md section 4 B5)	Deferred

8. Evidence References

- /tmp/alai/bilko-infra-team/petter-synthesis.md — 5-agent root-cause synthesis (Petter Graff)
- /tmp/alai/bilko-infra-team/devops1-cloudrun.md — Cloud Run compute/scaling layer audit (FlowForge)
- /tmp/alai/bilko-infra-team/devops2-data.md — Cloud SQL / HikariCP / gcsfuse / networking audit (FlowForge)
- /tmp/alai/bilko-infra-team/dev-applayer.md — Application layer audit (CodeCraft / Martin Kleppmann lens)
- /tmp/alai/bilko-infra-team/test-observability.md — Observability gaps + Proveo validation plan (Angie Jones)
- /tmp/evidence-103134/rebase-on-entra.md — Fix branch: PermissionService finding, rebase evidence, build/test results
- /tmp/evidence-103134/proveo-stage-verdict.json — Stage validation machine-readable verdict (Proveo, 2026-06-08T09:54:10Z)

- /tmp/evidence-103134/proveo-stage-validation.md — Stage validation human-readable report
- /tmp/alai/p2p-pairing-evidence/john-7fedd67f-freeze-fix-coordination-20260608.md — Cross-session coordination with Entra workstream

---

Revision #1
Created 2026-06-08 11:02:51 UTC by John
Updated 2026-06-08 11:02:51 UTC by John