APEX CLOUD SERVICES — INCIDENT RESPONSE

RUNBOOK: Nimbus Gateway Service

Document ID: RNB-NGS-402 — Version 3.4.1

1 System Overview

The **Nimbus Gateway Service (NGS)** serves as the unified ingress point and API gateway for all external-facing client applications in the Apex Cloud ecosystem. It manages rate limiting, authentication verification, dynamic routing, and payload inspection.

NGS relies on a high-speed distributed cache (Vertex Memory Grid) to validate authentication tokens and throttle traffic. If the memory grid becomes unreachable or experiences high read/write latency, the gateway falls back to the database tier (Synapse Distributed Store), which can quickly saturate database connections and cause cascading failures across the downstream microservices.

The incident response procedures in this runbook are designed in alignment with standard distributed reliability practices described by Jenkins [1] and failover mechanics discussed in [2].

1.1 Operational Metadata

Below is the vital metadata for the Nimbus Gateway Service. SREs should check these channels first during an incident.

System Code:	NGS-402	**Tier Rating:**	Tier-1 (Critical)
Primary Owner:	Meridian Traffic Team	**Lead Architect:**	Elena Rostova
Slack Channel:	#ops-nimbus-gateway	**PagerDuty Service:**	NGS-Ingress-Cluster
Primary Database:	Synapse Store (v12.4)	**Cache Cluster:**	Vertex Memory Grid

Important Notice: Service Level Objectives (SLOs)

This runbook is a critical document. During an incident, the Target Time to Acknowledge (TTA) is **5 minutes**, and the Target Time to Restore (TTR) is **20 minutes**. Failure to resolve NGS cache saturation within 20 minutes triggers an automatic escalation to the VP of Engineering.

2 Alert-to-Action Mapping

The following mapping specifies the entry-point remediation procedures for common alerts triggered by the Prometheus monitor. Locate the alert name in the table below and execute the corresponding remediation steps in Section 3.

Prometheus Alert Name	Severity	Status Page	Initial Diagnostics & Remediation Link
`NGS_Cache_Saturation`	**CRITICAL**	Red	Cache memory usage $>$ 92%. Perform memory purge and scaling. See Sec. 3.1.
`NGS_Latency_P99_Spike`	**WARNING**	Yellow	Response latency $>$ 250ms. Check downstream DB queue lag. See Sec. 3.2.
`NGS_HTTP_5xx_Rate`	**CRITICAL**	Red	Gateway error rate $>$ 4%. Check auth routing pod logs. See Sec. 3.3.
`NGS_DbPool_Exhausted`	**WARNING**	Yellow	Connection pool saturation $>$ 85%. Scale database connection limits. See Sec. 3.4.

2.1 Standard Diagnostics Commands

Before taking remediation steps, run these standard diagnostic scripts to collect system state information. This data will be automatically attached to the incident ticket.

Diagnostic Script 1: Cluster Health Status Check

```
# Check pod logs and query node status on the Apex Cloud Kubernetes cluster
kubectl get nodes -o wide --selector=app=nimbus-gateway
kubectl describe pods -l app=nimbus-gateway | grep -E "Status:|Reason:|Message:"
```

Diagnostic Script 2: Cache Performance Metrics

```
# Retrieve cache hits/misses ratio from the Vertex Memory Grid CLI
vertex-cli -h cache-cluster.meridian.internal -p 6379 info stats | grep -E "keyspace_|hit_rate"
```

3 Remediation Steps

This section contains step-by-step procedures to resolve alerts mapped in Section 2.

3.1 NGS Cache Memory Saturation

If the alert `NGS_Cache_Saturation` is active, follow these steps immediately to evacuate cache nodes and increase cluster capacity.

DANGER: DO NOT RESTART ENGINE DIRECTLY

Restarting the entire Vertex Memory Grid cluster under load will cause an immediate cache stampede, routing all auth requests to the Synapse Store database and knocking the database offline. Always scale first, then run target evictions.

1. **Check Eviction Policy:** Run the command below to ensure the eviction policy is set to `volatile-lru`. If it is set to `noeviction`, change it immediately.

Check Eviction Policy

```
vertex-cli -h cache-cluster.meridian.internal config get maxmemory-policy
```

2. **Set Eviction Policy (if not volatile-lru):** Execute the following configuration change to allow key eviction under memory pressure:

Change Eviction Policy

```
vertex-cli -h cache-cluster.meridian.internal config set maxmemory-policy volatile-lru
```

3. **Scale Memory Cluster Nodes:** Scale the cache deployment up by 4 nodes in the Vertex Cloud Control Plane:

Scale Memory Pods

```
kubectl scale statefulset/vertex-cache-nodes --replicas=8 -n ops-ingress
```

4. **Verify Cluster Re-balancing:** Monitor the re-balance script to verify that keys are being distributed to new nodes:

Monitor Cluster Status

```
vertex-admin-tool --cluster-status --check-balance
```

3.2 Latency P99 Spike Response

A spike in latency is typically caused by connection queuing or unindexed lookup operations.

1. **Analyze Dynamic Routing Tables:** Check if dynamic routing tables contain stale endpoints:

Route Validation

```
curl -s http://nimbus-gateway-admin.internal/v1/routes/health | jq '.bad_endpoints'
```

Action: If any endpoint reports a status other than `UP`, prune it using the route administration API:

Prune Unhealthy Endpoint

```
curl -X POST -d '{"status":"PRUNE"}' http://nimbus-gateway-admin.internal/v1/routes/prune
```

2. **Check Thread Saturation:** Identify thread utilization across the ingress proxy pods:

Thread Utilization Diagnostic

```
kubectl exec -it deployment/nimbus-gateway-proxy -n ops-ingress -- top -b -n 1
```

3.3 Gateway 5xx Error Mitigation

When 5xx errors spike, client requests are being rejected or dropping at the ingress controller.

IMPACT WARNING: Dynamic Payloads

If the error code is 503, the gateway is failing to reach downstream microservices. Do NOT adjust gateway routing rules unless the downstream microservice status page is verified as Green.

1. **Check Gateway Ingress Logs:** Parse the recent access logs for 502/503 errors and look for upstream timeout markers:

Filter Ingress Error Logs

```
kubectl logs -l app=nimbus-gateway --tail=500 | grep -E "HTTP/1.1\" 50[23]" | head -n 20
```

2. **Enable Temporary Rate Throttling:** If downstream services are saturated, apply a global 15% rate limit to non-premium API keys to allow the downstream systems to recover:

Apply Rate Limiting Override

```
curl -X PUT -H "Content-Type: application/json" \
  -d '{"throttle_ratio": 0.85, "target_tier": "standard"}' \
  http://nimbus-gateway-admin.internal/v1/throttling/override
```

3.4 Database Connection Pool Exhaustion

If the database connection pools are saturated, the system will experience cascading request timeouts.

1. **Inspect Open Database Connections:**

Retrieve Database Pool Usage

```
synapse-cli -h db-primary.meridian.internal -u admin -p --query "SHOW PROCESSLIST"
```

2. **Kill Runaway Queries:** If any query has been running for more than 10 seconds, terminate it:

Terminate Long Running Queries

```
synapse-cli -h db-primary.meridian.internal -u admin -p --query "CALL kill_queries_older_than(10)"
```

3. **Scale Connection Pools:** Adjust the gateway's max pool size configurations dynamically:

Dynamically Scale Database Connections

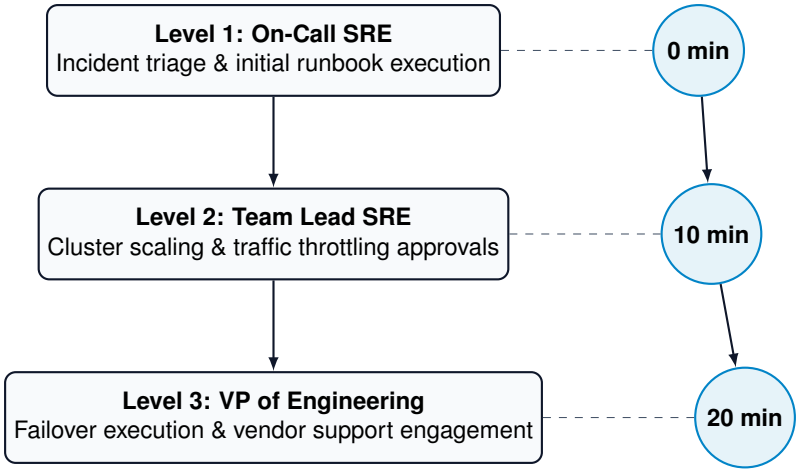
```
kubectl set env deployment/nimbus-gateway-proxy DB_MAX_CONNECTIONS=150 -n ops-ingress
```

4 Escalation Ladder & Contacts

When an incident cannot be resolved within the target restore time (TTR) or requires additional authorization, escalate immediately using the ladder below.

4.1 Visual Escalation Path

The diagram below illustrates the escalation milestones based on elapsed time since the initial alert trigger.



4.2 Contact Directory

Use the directory below to contact the team members on duty. Do NOT contact Level 2 or Level 3 unless the escalation milestones in the path above have been reached.

Escalation Role	Name	Incident Channel	PagerDuty Alias
Level 1: On-Call SRE	Julian Vance	#ops-nimbus-gateway	`@vance-oncall`
Level 2: SRE Team Lead	Sarah Jenkins	#ops-escalations	`@sjenkins-lead`
Level 3: VP of Engineering	Elena Rostova	#ops-emergency	`@erostova-vp`
Database Administrator (DBA)	Marcus Vance	#ops-db-alerts	`@mvance-dba`
Vertex Cluster Support	(Vendor Support)	Hot-line: +1-555-0199	`vertex-priority`

Incident Close-Out Procedure

Once the system is restored to green status, the Level 1 SRE is responsible for:

1. Closing the active incident ticket on PagerDuty.
2. Running the automated diagnostic capture script and posting the summary to the incident channel.
3. Scheduling a post-mortem review within 24 hours of the incident.

References

[1] Sarah Jenkins. *Distributed Systems Reliability: Engineering Ingress at Scale*. Meridian Academic Press, Meridian City, 2024.

[2] Elena Rostova and Marcus Vance. Failover mechanics and db connection management under cache saturation. Technical Report SRL-TR-2025-11, Synapse Research Labs, 2025.

[3] Julian Vance. Dynamic load balancing and routing algorithms for high-throughput gateways. *Journal of Vertex Engineering*, 14(2):45–58, 2023.