Incident Post-Mortem & Monitoring Improvement Proposal

Incident ID: INC-EX3 | **Date:** December 28, 2025 | **Duration:** 45 minutes

Author: Nick Hinds | **Severity:** High

Executive Summary

Production API experienced severe performance degradation: response times increased from 200ms to 3000ms (15x) and 5% of requests returned 504 Gateway Timeout errors during peak traffic.
Root cause: Database connection pool exhaustion triggered by 3x traffic spike from unannounced marketing campaign. Impacted ~75,000 requests with $12,000 revenue loss.

Key Learnings: Inadequate monitoring prevented early detection (MTTD: 13 minutes). No alerts existed for connection pool utilization, which would have provided 10-15 minute advance warning.

Timeline & Root Cause Analysis

Time	Event	Finding
14:15	Latency increases to 800ms	Traffic spike begins
14:28	Alert triggered (P50 >1000ms)	13-minute detection gap
14:35	DB connections at 100% (50/50)	Root cause identified
14:38	3x traffic from campaign URLs	Marketing launch, no notification
14:42	Pool increased 50→150	Emergency mitigation
15:00	Full recovery (250ms, 0% errors)	MTTR: 45 minutes

Contributing Factors: Insufficient connection pool sizing • No circuit breaker • Missing auto-scaling triggers • Communication breakdown between teams • No load testing at 3x capacity

Critical Monitoring Gaps Identified

Gap	Impact During Incident	Proposed Solution
No connection pool alert	No advance warning despite 10-15 min runway	Alert at 80% (warn), 90% (critical)
High latency threshold	Alert at 1000ms too late (3000ms peak)	Alert at 400ms (2x), critical at 600ms
No error rate monitoring	5% 504 errors undetected by alerts	Alert at 1% error, critical at 2.5%
Missing traffic anomaly	3x spike went unnoticed	Anomaly: 2 std dev threshold

Remediation Plan

Immediate Actions (Completed): ✓ Increased DB connection pool: 50 → 150 • ✓ Scaled instances: 4 → 12 • ✓ Added temporary connection pool monitoring

Short-term (1-2 weeks) - Priority P0/P1

Priority	Action	Owner	Due
P0	Deploy critical alerts (pool, latency, errors, traffic)	SRE	Dec 20
P0	Implement circuit breaker for DB connections	Backend	Dec 22
P1	Configure auto-scaling on pool metrics (>75% → scale)	Platform	Dec 27
P1	Deploy primary operations dashboard	SRE	Dec 27
P1	Establish campaign launch communication process	Eng+Marketing	Jan 15

Medium-term (1 month) - P2: Connection pool optimization with query timeouts • Monthly load testing (2x, 3x, 5x capacity) • Business impact dashboard

Monitoring Dashboard & Alert Proposal

Dashboard: API Health & Performance (Primary Operations)

```
Request Metrics: Request Rate • Success Rate • Error RateLatency Distribution: P50 (target
<250ms) • P95 (target <500ms) • P99 (target <1000ms)Database Health: Connection Pool % ⚠ •
Active Queries • Connection Wait TimeInfrastructure: Instance Count (auto-scaling) • CPU (alert
>75%) • Memory (alert >85%)
```

Alert Configurations

Critical Alerts (PagerDuty):

Alert	Critical Threshold	Warning Threshold
1. High Error Rate (5xx)	> 2.5% for 2 min	> 1% for 3 min
2. Gateway Timeouts (504)	> 10 in 5 min	> 5 in 5 min
3. API Latency Spike	P95 > 600ms for 3 min	P95 > 400ms for 5 min
4. DB Connection Pool ⭐	> 90% for 2 min	> 80% for 5 min
5. Traffic Anomaly	—	> 2 std dev for 5 min

Alert Routing: Critical → PagerDuty (on-call) + #incidents (Slack) • Warning → #alerts (Slack) • Escalation: On-call → Secondary (+5 min) → Manager (+10 min)

Implementation & Success Metrics

4-Phase Rollout (4 weeks): Week 1: Deploy 5 critical alerts + PagerDuty integration • Week 2: Deploy primary dashboard, train on-call team • Week 3: Add warning alerts + business metrics dashboard • Week 4: Capacity planning dashboard + monthly review process

Expected Outcomes (3-month targets)

Metric	Current	Target	Impact
MTTD	13 min	< 5 min	60% faster detection
MTTR	45 min	< 30 min	33% faster resolution
Proactive Detection	20%	> 80%	Prevent customer impact
Customer Incidents	4/month	< 2/month	50% reduction

Cost Analysis

```
Additional Monitoring Costs:      $256/month
Datadog APM expansion:            $124/month
Log ingestion (200GB):            $20/month
Custom metrics:                   $30/month
PagerDuty (2 users):              $82/month
ROI: Prevent 1 incident/quarter = $12,000/year saved vs. $3,072/year additional cost = 290% ROI
```

Preventive Measures

Technical: Circuit breaker implementation • Connection pooling proxy (PgBouncer) • Rate limiting • Service mesh
Process: Pre-launch infrastructure checklist • Quarterly game day exercises • Incident response playbooks
Organizational: Monthly capacity reviews • Cross-team launch coordination • SRE embedding in product teams

Key Takeaways

What Went Well: Fast root cause identification (20 min) • Effective recovery actions • Clear team communication
What Could Improve: Proactive monitoring for early detection • Capacity planning collaboration • Auto-scaling configuration
Critical Success Factor: The proposed connection pool alert would have detected this issue 10-15 minutes before customer impact, preventing 75,000 failed requests and $12,000 in lost revenue.

Next Steps: Stakeholder review (Jan 5) → Budget approval (Jan 7) → Phase 1 deployment (Jan 9)
Approvals Required: Engineering Manager, SRE Lead, VP Engineering