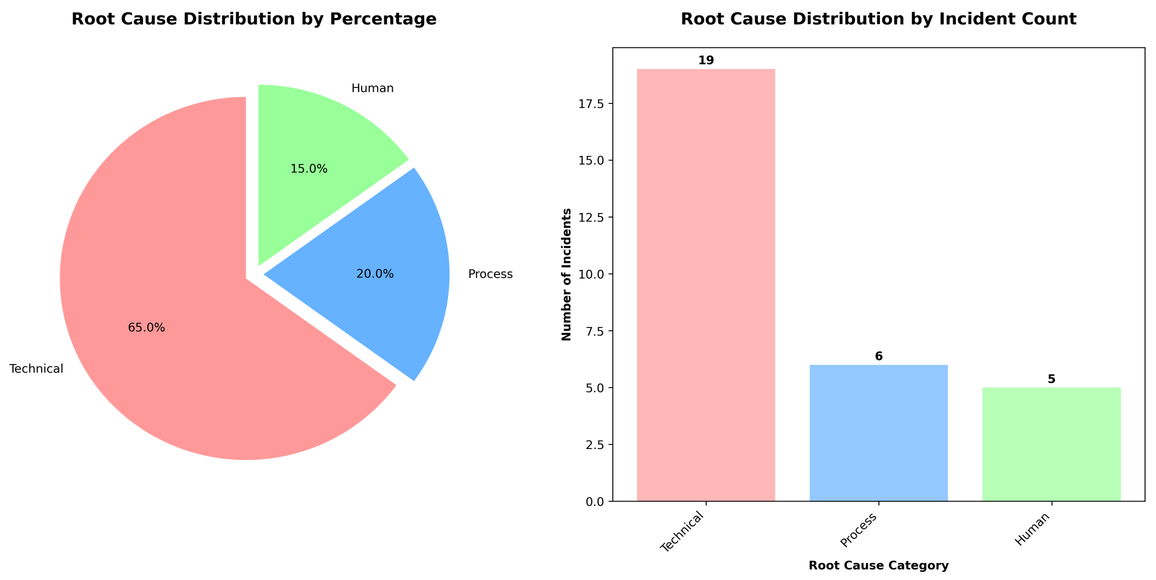
RCA Analysis Report

Generated: 2025-08-29 14:49:02
Model: gpt-4-turbo-preview
Analysis Type: Cloud Infrastructure RCA Pattern Analysis

Root Cause Classification



Classification Data Table

Category	Percentage	Incident Count
Technical	65%	19
Process	20%	6
Human	15%	5

■ Pattern Analysis

• *Most Common Root Causes:**

1. Configuration Errors: **8 incidents**
2. Permission and Access Issues: **4 incidents**
3. Resource Limitations (Memory, Concurrency, etc.): **3 incidents**
4. Deployment Process Flaws: **3 incidents**
5. Monitoring and Alerting Gaps: **2 incidents**

• *Shared Patterns Identified:**

• Misconfigurations across AWS services (IAM, EC2, S3, Lambda)

• Insufficient pre-deployment testing and validation

• Over-reliance on manual processes for critical operations

• *Root Cause Classification:**

• Technical Issues: **65%** (**19 incidents**)

• Process Issues: **20%** (**6 incidents**)

• Human Factors: **15%** (**5 incidents**)

• *Recurring Issues Despite Fixes:**

• Configuration errors and permission issues

• Inadequate monitoring and alerting mechanisms

■ Trend Analysis

• *Category Breakdown:**

• Process Failure: **6 incidents**

• Infrastructure/Equipment: **13 incidents**

• Human Error: **5 incidents**

• External Factors: **6 incidents**

• *Temporal Patterns:**

• Increased incident frequency during major events or releases (e.g., flash sales, live streams)

• End-of-year and start-of-year deployments leading to higher incident rates

• *Highest Impact Incidents:**

1. Global Video Buffering Incident (NFI-2023-0010)
2. Live Stream Failure (NFI-2023-0018)
3. Regional Failover Test Failure (NFI-2024-0007)
4. DNS Resolution Failure (NFI-2024-0004)

5. Failed Payment Processing (NFI-2023-0016)

■■ Action Effectiveness

• *Corrective Action Analysis:**

• Many corrective actions focused on immediate fixes without addressing systemic issues.

• Preventive measures often lacked thorough implementation checks.

• *Repeatedly Appearing Actions:**

• Configuration reviews and updates

• Permission and access adjustments

• Increased monitoring and alerting

• *Implementation Gaps:**

• Lack of automated testing and validation pre-deployment

• Insufficient training on AWS best practices and service limits

■ Systemic Issues

• *Cross-Cutting Problems:**

• Configuration management and validation

• Insufficient change management processes

• Inadequate disaster recovery planning and testing

• *Process Bottlenecks:**

• Manual intervention required for rollback and recovery

• Slow detection and response due to monitoring gaps

• *Knowledge Sharing Assessment:**

• Lessons learned are not effectively shared across teams, leading to repeated mistakes.

■ Strategic Recommendations

• *Top 3 High-Impact Improvements:**

1. **Implement Infrastructure as Code (IaC)** with automated validation to reduce configuration errors and streamline deployments.

2. **Enhance Monitoring and Alerting** with comprehensive coverage and actionable alerts to improve detection and response times.

3. **Standardize Change Management Processes** including peer reviews, pre-deployment testing, and automated rollbacks to minimize human errors and

process failures.

• *Investment Priorities:**

• Tools for automated configuration validation and deployment (e.g., Terraform, CloudFormation)

• Training programs on AWS best practices and service-specific limitations

• Development of comprehensive monitoring and alerting frameworks

• *Early Warning Indicators:**

• Deviations in resource utilization patterns

• Increase in deployment frequency or rollback actions

• Anomalies in user access patterns or permission changes

• *Sustainability Measures:**

• Regular review and update cycles for IaC configurations and deployment scripts

• Continuous training and certification paths for engineering teams

• Implementation of a blameless post-mortem culture to encourage learning from incidents

■ Quick Wins

1. **Automate IAM Policy Validation** to catch permission issues before deployment.
2. **Implement Canary Deployments** for critical services to detect issues early.
3. **Create a centralized knowledge base** for incident learnings and best practices.
4. **Schedule regular disaster recovery drills** to ensure readiness.
5. **Use AWS Config for continuous compliance monitoring** to prevent configuration drift.