**Spring Cloud Zuul 组件\_1.4.7版本\_运维规范（V1.0）**

**基础技术中心**

**2025年8月**

# 修订记录

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **编号** | **日期** | **描述** | **版本** | **作者** | **审核** | **发布日期** |
| 1 | 2025-08-17 | 建立文档 | 1.0 | 汤王 |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |

如无中国建设银行的书面许可，任何人都无权复制或利用。

**目 录**

[修订记录 2](#_Toc6409)

[1. 文档说明 4](#_Toc1821)

[2. 日常巡检规范 4](#_Toc1916)

[3. 日志管理要求 5](#_Toc32507)

[4. 备份策略 5](#_Toc25942)

[5. 监控与告警设置 6](#_Toc26546)

[6. 变更管理流程 7](#_Toc11397)

[7. 应急演练计划 7](#_Toc30147)

[8. 权限管理与安全要求 8](#_Toc8339)

# 文档说明

本文档是 Spring Cloud Zuul API 网关集群的日常运维标准手册，旨在为运维和 SRE 团队提供标准化的操作指引。 本规范的目标是确保网关服务的稳定性、安全性和高可用性，涵盖日常巡检、监控告警、应急处理等方面。

# 日常巡检规范

日常巡检分为每日健康检查和每周深度分析。

* **每日健康检查 (Daily Check):**
* **服务状态**: 确认所有 Zuul 节点的 Java 进程均处于 active (running) 状态。   
  systemctl status zuul-gateway.service
* **健康检查端点**: 检查每个节点的 /actuator/health 端点，确保返回 {"status":"UP"}。
* **错误日志**: 快速浏览应用日志，检查是否存在大量的 ERROR 或 WARN 级别的异常日志。
* **核心监控仪表盘**: 查看 Grafana 核心仪表盘，关注以下指标有无异常波动：
  + 请求量 (QPS)
  + P99 响应延迟
  + 5xx 错误率
* **每周深度分析 (Weekly Analysis):**
* **资源使用趋势**: 分析 JVM 堆内存、CPU 使用率的周度趋势，判断是否存在缓慢的内存泄漏或性能瓶颈。
* **磁盘空间**: 检查日志所在分区的磁盘使用率，确保其低于 80%。
* **GC (垃圾回收) 行为**: 分析 GC 活动的频率和耗时，评估 JVM 调优效果。
* **下游服务延迟**: 分析网关监控中记录的对各下游服务的平均响应时间，识别潜在的慢服务。

# 日志管理要求

* **日志级别**: 生产环境默认日志级别为 INFO。禁止在线上环境开启 DEBUG 级别，除非正在进行故障排查。
* **日志滚动**: 应用日志必须按天滚动，防止单个日志文件过大。
* **日志保留策略**:
  + 节点本地磁盘保留最近 7 天的日志文件。
  + 所有日志必须实时采集到中央日志平台（如 ELK、Loki），并在平台上至少保留  
    **90天**。
* **敏感信息保护:** 严禁在日志中记录任何敏感信息，包括但不限于用户密码、密钥、Token、身份证号等。

# 备份策略

Zuul 作为无状态网关，其备份主要集中在配置和应用版本上。

* **备份内容**:
  + **应用版本**: 所有经过测试、准发布的 JAR 包都必须归档在制品仓库（如 Nexus, Artifactory）中，并附带版本号。
  + **应用配置**: 所有的 application.yml、bootstrap.yml 等配置文件，必须存储在版本控制系统（如 Git）中进行管理。
* **备份频率**:
  + 应用配置：每次变更后都应提交到 Git。
  + 应用版本：每次构建成功后自动归档。
* **恢复方式**: 服务恢复通过 CI/CD 流水线实现，即从 Git 拉取指定版本的配置，从制品仓库下载对应版本的 JAR 包进行部署。

# 监控与告警设置

通过 Prometheus + Grafana + Alertmanager 体系对网关集群进行全方位监控。

* **核心监控指标:**
* **JVM 指标**: 堆内存使用率、GC 次数与耗时、线程数。
* **应用指标**:
  + http\_server\_requests：请求 QPS、响应延迟、HTTP 状态码分布（2xx/4xx/5xx）。
  + hystrix\_command\_event\_total：熔断器状态（成功、失败、超时、熔断）。
* **系统指标**: CPU 使用率、内存使用率、网络 I/O、磁盘 I/O。
* **告警阈值设定:**

| **告警级别** | **告警项** | **阈值** | **描述** |
| --- | --- | --- | --- |
| P1 (紧急) | 节点存活 | 节点健康检查失败 > 1 分钟 | 服务可能已宕机 |
|  | 5xx 错误率 | > 5% 持续 3 分钟 | 大量服务端错误，影响核心业务 |
|  | P99 响应延迟 | > 2s 持续 5 分钟 | 用户体验严重下降 |
| P2 (警告) | JVM 堆内存 | > 90% 持续 10 分钟 | 可能即将发生 OOM |
|  | CPU 使用率 | > 80% 持续 15 分钟 | 资源消耗过高，存在性能风险 |
|  | Hystrix 熔断 | 单个依赖服务的熔断器打开 | 下游服务异常，可能影响部分功能 |
| P3 (通知) | 磁盘使用率 | > 85% | 日志盘空间不足，需及时清理 |

# 变更管理流程

所有线上变更操作，包括配置修改、版本升级、节点扩缩容，都必须遵循标准的变更管理流程。

1. **创建变更单**: 在变更管理系统（如 JIRA）中创建变更请求，详细说明变更原因、内容、风险和回滚方案。
2. **技术评审**: 变更方案需由至少两名资深工程师进行评审。
3. **预生产验证**: 变更必须先在预生产环境部署，并由测试团队验证通过。
4. **变更审批**: 获得技术负责人或变更审批委员会（CAB）的批准。
5. **发布执行**: 在批准的变更窗口期内，由授权人员执行变更操作。
6. **变更后验证**: 变更完成后，执行业务验证，并持续监控至少 30 分钟。
7. **关闭变更单**: 确认服务稳定后，关闭变更单。

# 应急演练计划

为检验和提升运维团队的应急响应能力和系统的健壮性，定期组织应急演练。

|  |  |  |  |
| --- | --- | --- | --- |
| 演练项目 | 演练频率 | 演练内容 | 目标 |
| **节点故障切换** | 每季度 | 随机 kill -9 杀掉一个 Zuul 节点 | 验证 LB 是否能自动摘除故障节点，业务无感知 |
| **下游服务熔断** | 每半年 | 手动停止一个非核心下游服务 | 验证 Zuul 的 Hystrix 熔断或 Sentinel 降级是否生效 |
| **配置错误回滚** | 每半年 | 模拟发布一个错误的配置文件 | 演练配置快速回滚流程，衡量恢复时间（RTO） |
| **灾备恢复** | 每年 | （若有）模拟主数据中心故障，切换到灾备中心的流程 | 验证跨中心切换方案的可行性 |

# 权限管理与安全要求

* **最小权限原则**: 严格遵循最小权限原则，为运维人员分配完成工作所必需的最小权限。
* **账户管理**:
  + Zuul 应用必须使用独立的、低权限的系统用户（如 zuul-user）运行。
  + 禁止使用 root 用户部署和运行应用。
  + 所有服务器登录账户必须使用强密码，并定期更换。
* **访问控制**:
  + 生产服务器的 SSH 访问必须通过堡垒机进行。
  + Spring Boot Actuator 的管理端口必须配置访问控制，或部署在内部管理网络，禁止暴露在公网。