**Spring Cloud Zuul组件\_1.4.7版本\_版本升级方案（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 | 2025-08-18 | 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](#_Toc31884)

[1. 背景 4](#_Toc26246)

[1.1. 适用范围 4](#_Toc967)

[1.2. 升级背景 4](#_Toc4450)

[2. 升级影响性评估 4](#_Toc5034)

[3. 升级方案 5](#_Toc14839)

[3.1. 升级要求 5](#_Toc29777)

[3.2. 升级方式 5](#_Toc24435)

[3.3. 升级操作 5](#_Toc10667)

[3.4. 注意事项 7](#_Toc22937)

# 背景

## 适用范围

本方案适用于所有在生产环境中使用 Spring Cloud Zuul 组件的应用，特别是当前依赖版本为 1.4.6.RELEASE 或更早版本的系统。本文档将提供详细的操作指引，以确保平稳、安全地升级至 1.4.7.RELEASE 版本。

## 升级背景

本次升级的主要目标是**修复严重的安全漏洞** 。

* **漏洞修复**：官方在 Spring Cloud Zuul 1.4.7.RELEASE版本中修复了一个编号为 **CVE-2019-3799** 的高危路径穿越漏洞。该漏洞允许未经授权的攻击者通过构造恶意请求来访问网关服务器上的敏感文件，对系统信息安全构成严重威胁。
* **稳定性提升**：作为官方发布的补丁版本，1.4.7.RELEASE 包含了一些稳定性修复，有助于提升网关服务的整体健壮性。

# 升级影响性评估

* **服务中断**：
  + 本次升级涉及应用代码依赖变更，需要重新编译打包和部署，因此 **单个节点需要停机**。
  + 为保证业务连续性，生产环境 **必须采用滚动升级（Rolling Upgrade）** 方式进行，在升级期间可实现整个集群服务的 **无中断**。
* **数据风险**：
  + Spring Cloud Zuul 作为网关组件，本身是无状态的，不涉及数据存储和迁移。
  + 因此，本次升级无数据丢失或数据不一致的风险。
* **兼容性风险**：
  + 1.4.7.RELEASE 是针对 1.4.x 系列的补丁版本，API 与低版本（如 1.4.6）保持完全兼容，理论上不存在兼容性问题。
  + 尽管风险极低，但仍需在预生产环境进行完整的回归测试，以确保业务逻辑无任何影响。

# 升级方案

## 升级要求

* **环境要求**：必须具备一套与生产环境配置一致的、可独立进行测试的预生产环境（UAT 或 Staging）。
* **依赖组件**：Spring Cloud Zuul 1.4.7.RELEASE 属于 Spring Cloud Edgware.SR6 版本 ट्रेन, 其构建基于 Spring Boot 1.5.x。请确保您的项目使用的 Spring Boot 版本与之兼容。
* **代码与构建**：需要准备好应用服务的完整源代码，并确保CI/CD构建环境正常可用。

## 升级方式

本次升级通过修改项目构建文件（pom.xml）中的依赖版本来完成。

**推荐方式**：在生产集群中，采用 **滚动升级** 的方式进行部署，以确保服务的高可用性 。

## 升级操作

### 升级前准备工作检查

* **代码备份**：在开始修改前，备份当前分支的代码，或新建一个专门用于升级的分支。
* **应用备份**：备份当前在生产环境中运行的应用 JAR 包，以便在必要时快速回滚。
* **预生产验证**：确保新版本的代码已在预生产环境 **完成充分测试**，包括功能回归测试、性能压测和安全扫描。
* **监控确认**：确认监控系统（如 Prometheus, Zabbix）运行正常，能够清晰地观察到各节点升级前后的状态（CPU、内存、JVM、错误率等）。
* **发布通知**：提前通知所有相关方（开发、测试、运维、业务部门）本次升级的窗口期。

### 升级操作步骤描述

步骤一：修改项目依赖

1. 打开项目的根 pom.xml 文件。
2. 在 <properties> 标签中，将 spring-cloud.version 的值修改为 Edgware.SR6。  
   XML  
   <properties>  
    <spring-cloud.version>Edgware.SR6</spring-cloud.version>  
   </properties>
3. 执行 Maven 命令 mvn clean dependency:tree 确认 spring-cloud-starter-netflix-zuul 依赖的版本已更新为 1.4.7.RELEASE。

**步骤二：编译打包**

1. 在项目根目录下执行打包命令：  
   Bash  
   mvn clean package -DskipTests
2. 将生成的新版 JAR 包上传到服务器的应用发布目录。

**步骤三：执行滚动升级（以3节点集群为例）**

1. **升级 Node-1**:
   * 将 Node-1 从负载均衡（如 Nginx, F5）的后端服务列表中摘除，停止流量进入。
   * 停止 Node-1 上旧版本的 Zuul 应用进程。
   * 替换为新版本的 JAR 包，并启动应用。
   * 检查启动日志，确保无异常错误。通过 curl http://<node-1-ip>:port/actuator/health 确认服务健康状态为 UP。
   * 将 Node-1重新加入负载均衡服务列表，恢复流量。
   * 持续观察监控指标 5-10 分钟，确认服务稳定。
2. **升级 Node-2**:
   * 确认 Node-1 稳定后，对 Node-2 重复以上操作（摘除流量 -> 停旧版 -> 上新版 -> 验证 -> 恢复流量 -> 观察）。
3. **升级 Node-3**:
   * 确认 Node-2 稳定后，对 Node-3 重复以上操作。
4. **最终验证**：所有节点升级完毕后，观察整个集群的运行状态，确认业务访问完全正常。

### 失败回滚方案

如果在升级任何一个节点时出现以下情况，立即执行回滚：

* 应用启动失败。
* 启动后健康检查持续失败。
* 日志中出现大量异常。
* 加入负载均衡后，该节点的错误率或延迟显著升高。

**回滚步骤：**

1. 立即将问题节点从负载均衡中 **彻底移除**。
2. 停止该节点上失败的新版本应用。
3. 使用 **步骤 3.3.1 中备份的旧版本 JAR 包** 替换当前应用。
4. 重新启动旧版本应用，并进行健康检查。
5. 确认服务恢复正常后，将其重新加入负载均衡。
6. **终止后续节点的升级计划**，并立即组织技术人员根据日志和监控快照排查失败原因。

## 注意事项

* **禁止全量升级**：严禁在生产环境一次性停止所有节点进行升级，必须严格遵守滚动升级流程。
* **备份配置**：尽管本次升级不涉及配置文件变更，但在执行任何操作前，备份应用的 application.yml 或 bootstrap.yml 等配置文件依然是最佳实践 。
* **密切监控**：在整个升级窗口期以及升级完成后的24小时内，需要对网关集群的核心性能指标进行密切监控，以便及时发现潜在问题。