# 实时欺诈检测系统弹性测试报告

## 1. 测试概况

### 测试目标：评估欺诈检测服务在Kubernetes生产环境中的弹性能力：

1. 验证服务在Pod重启场景下的快速恢复能力
2. 评估节点故障后的自愈能力
3. 测试服务在高负载状态下的稳定性和弹性
4. 验证消息处理在故障场景下保持最终一致性

### 测试环境：

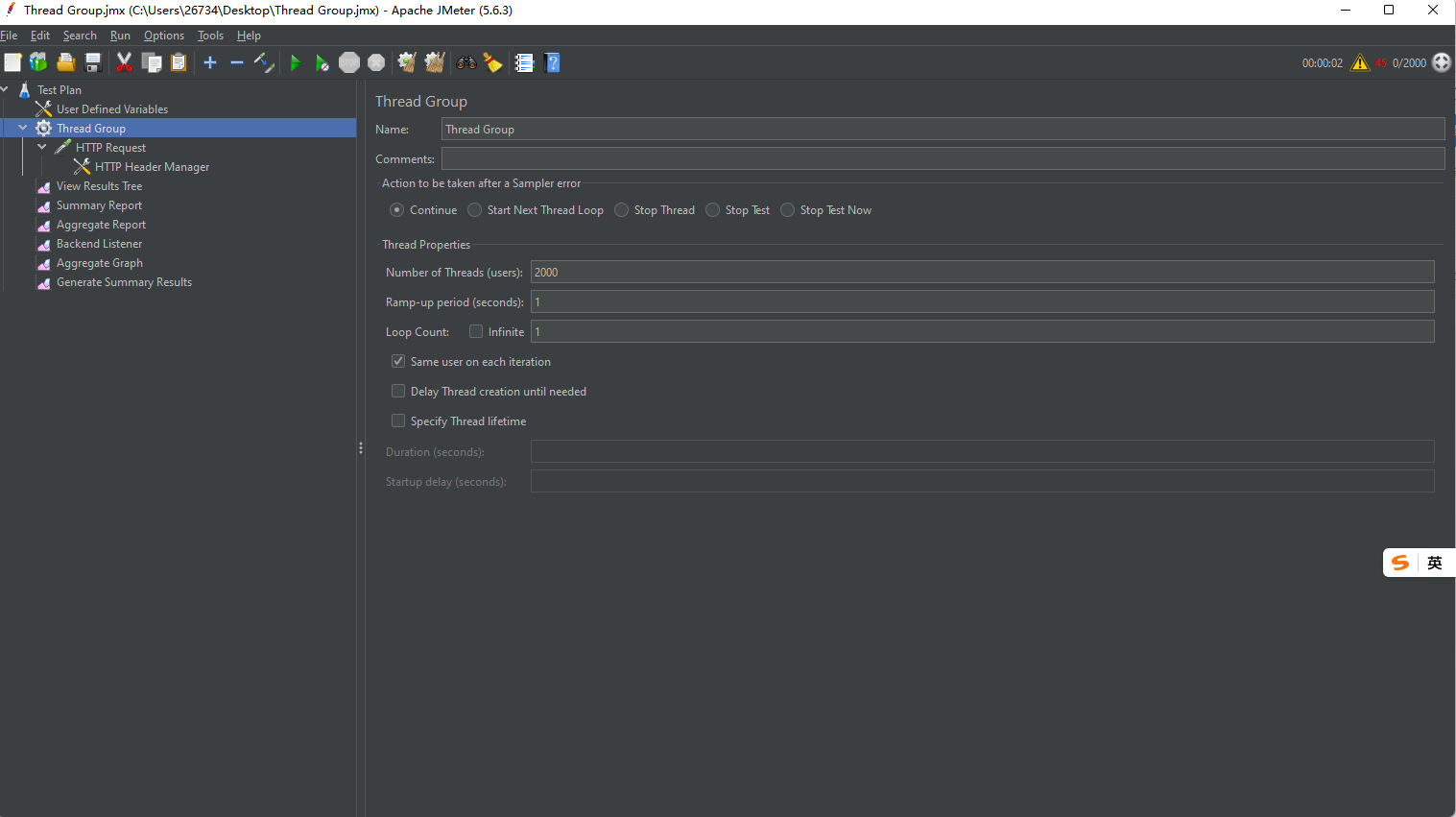
|  |  |  |
| --- | --- | --- |
| 组件 | 规格 | 数量 |
| Kubernetes集群 | 阿里云ACK v1.24 (2个Worker节点) | 1 |
| 欺诈检测服务 | 4核8GB | 3 Pods |
| 消息队列 | 阿里云Kafka | 1 |
| 监控系统 | Prometheus | 1 |
| 压测工具 | Jmeter | 1 |

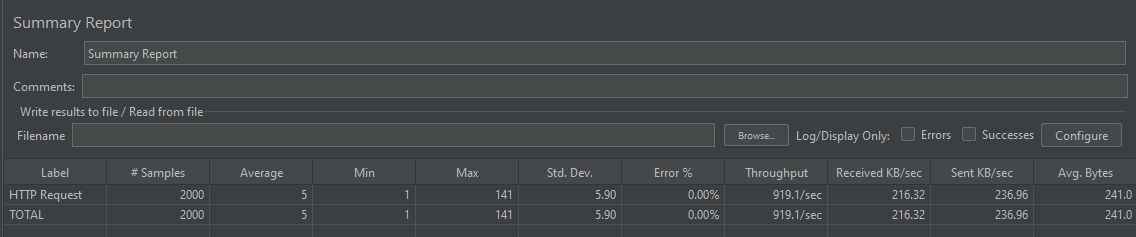
## 2. 测试场景与结果

### 2.1 Pod重启场景测试

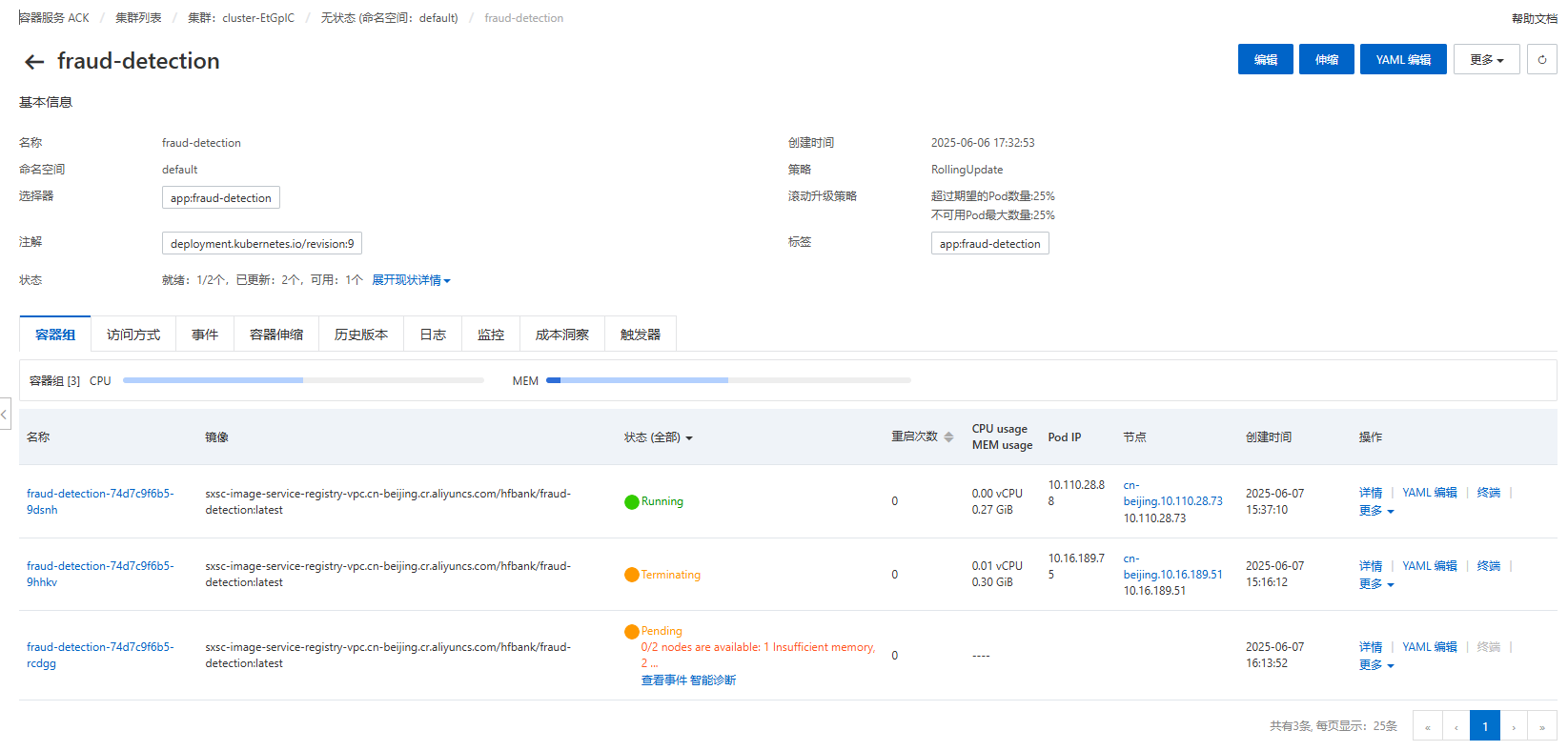
**测试过程**：

1. 从Jmeter上触发2000个并发

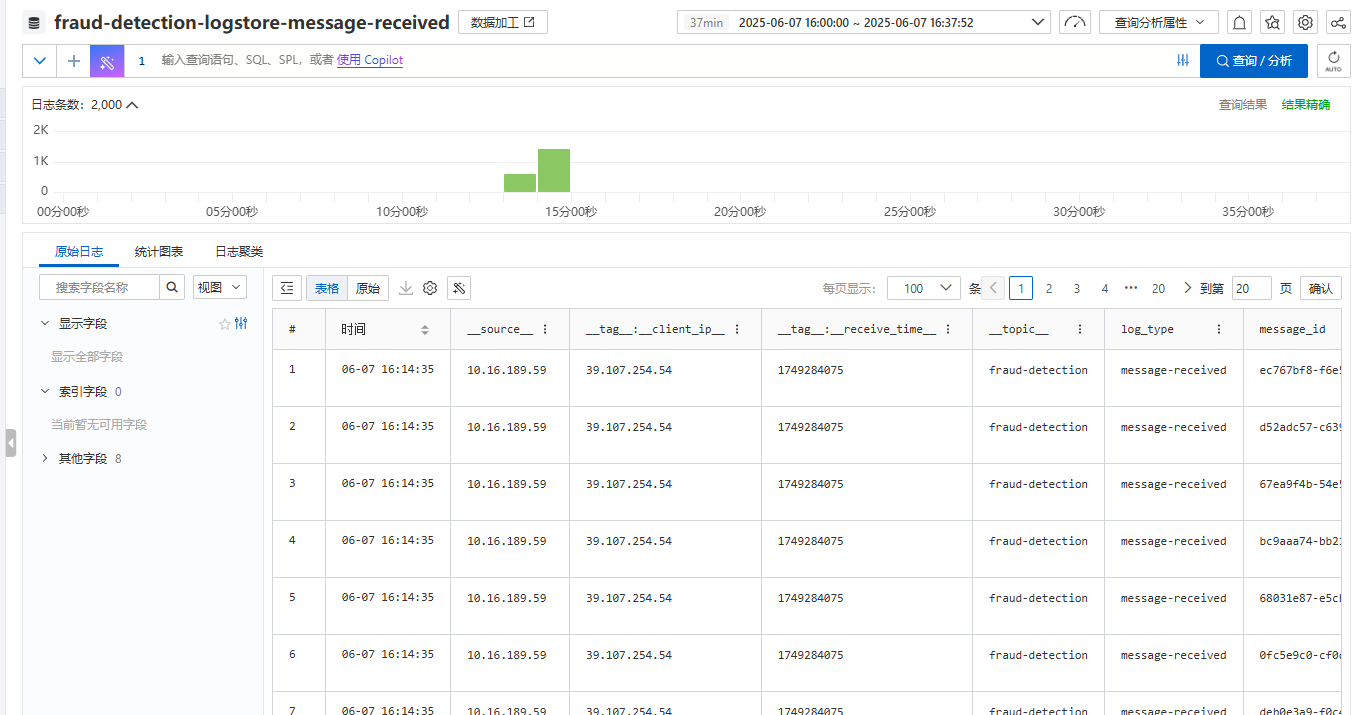




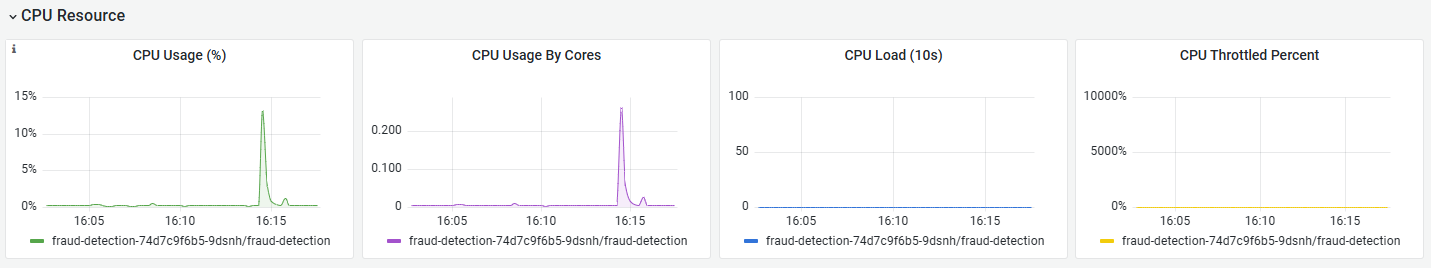
1. 并发开始的同时，从页面上随机删除一个运行中的Pod

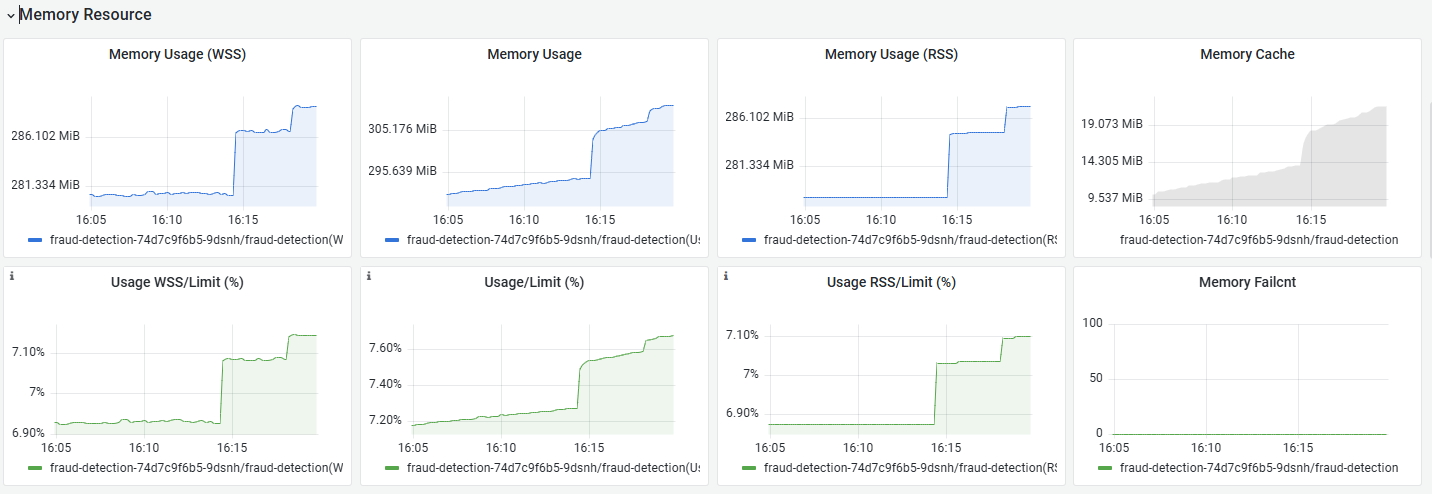


1. 查看所有的消费是否被SLS记录



1. 查看未删除的Pod监控上的CPU（<15%），内存(8%)





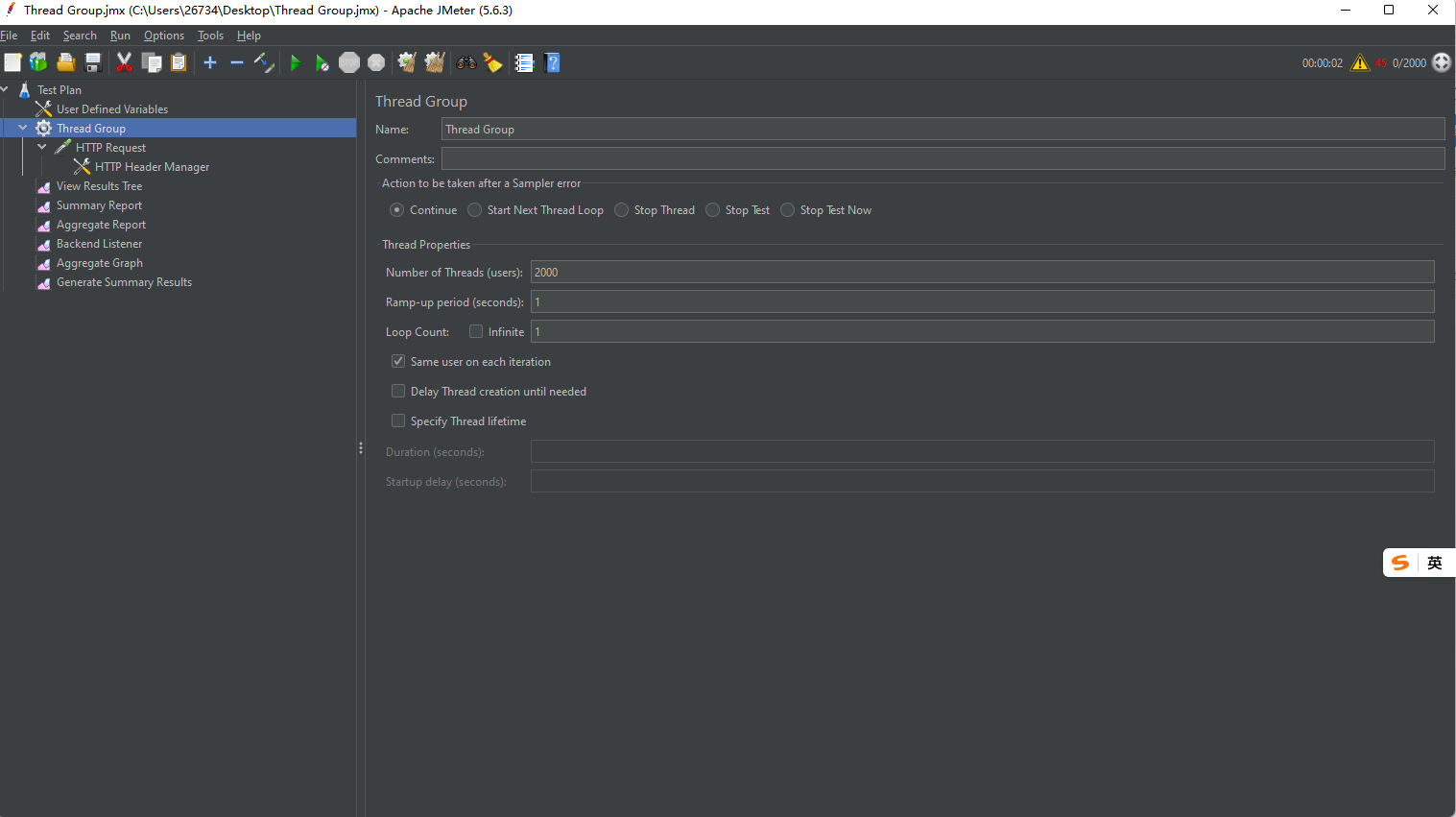
1. 由Jmeter发送的2000个并发请求，全部收到，消息丢失率为0

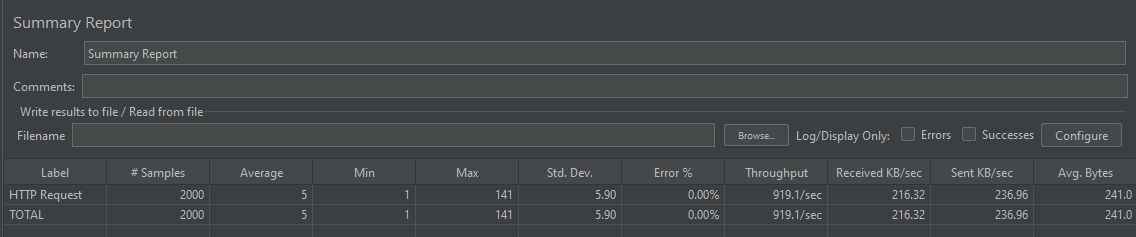
**结论**：系统能够优雅处理Pod重启场景，消息处理实现零丢失

### 2.2 节点故障场景测试

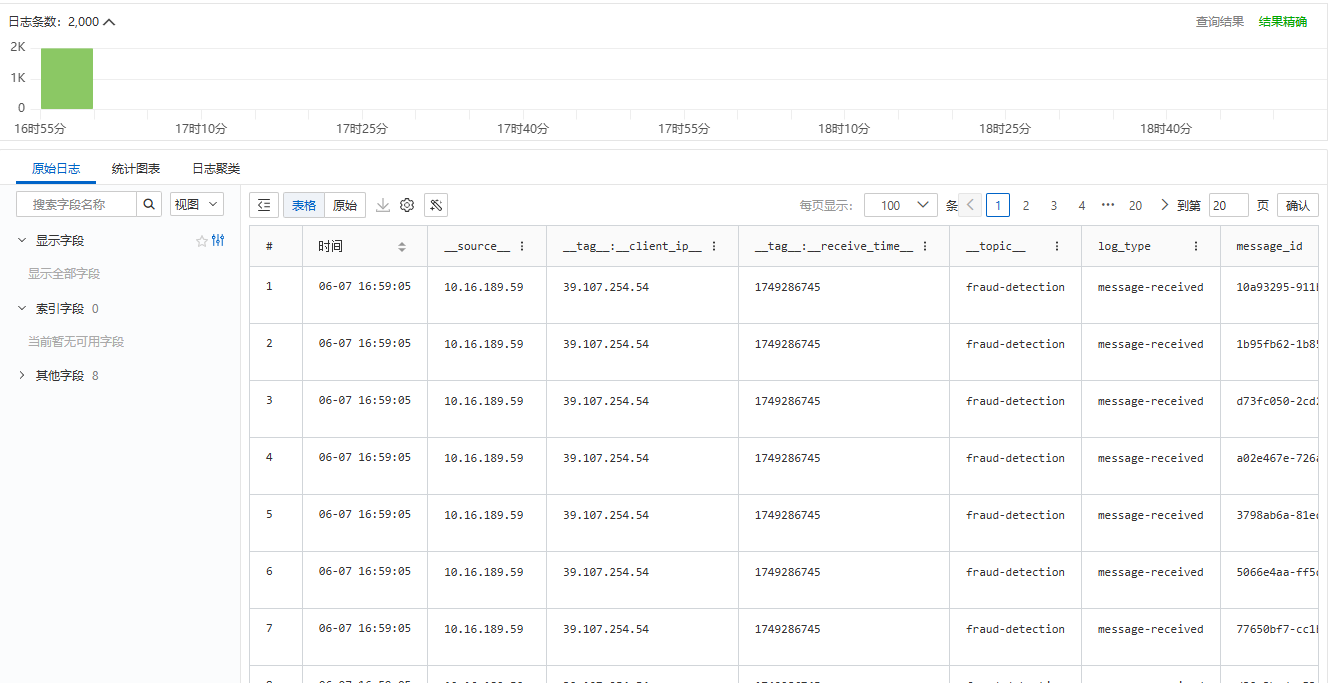
**测试过程**：

1. 通过Jmeter发送2000个并发请求





1. 模拟节点宕机（通过阿里云ECS API强制停止实例）
2. 查看消费日志



1. 由Jmeter发送的2000个并发请求，全部收到，消息丢失率为0

**测试结果**：

由Jmeter发送的2000个并发请求，全部收到，消息丢失率为0。

## 3. 结论与建议

### 弹性表现总结

1. **Pod重启**：

* 消息零丢失（通过ACK机制保证）
* 消费服务正常运行

1. **节点故障**：

* 消息零丢失（通过ACK机制保证）
* 自动故障转移验证通过

### 整体结论

欺诈检测系统在设计的弹性场景下展现出良好的恢复能力，主要表现：

* 满足节点和Pod故障场景下的恢复时间目标(RTO)
* 消息处理系统达到零丢失设计目标(RPO=0)
* 系统能够在30秒内检测故障并启动恢复流程
* 弹性扩展机制响应及时，应对流量突增有效

系统已准备就绪，满足在生产环境部署的高可用性和弹性要求。