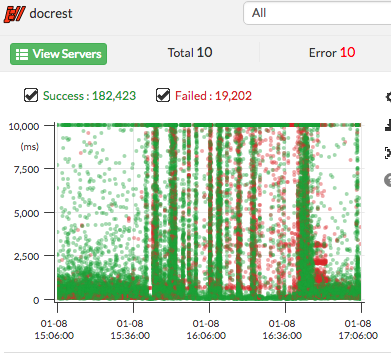
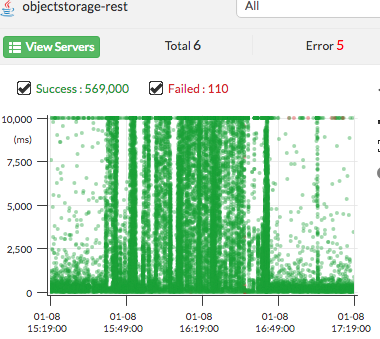
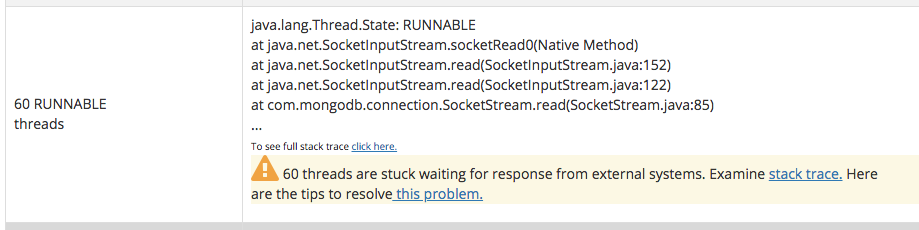
**[2019-01-07签到服务宕机总结](http://192.168.0.22/cms/pages/viewpage.action?pageId=20814664)**

#### 故障分析：

故障发生在下午15:30后，表现为最底层的objectstorage-rest响应缓慢，docrest依赖objectstorage缓慢且大量报错。

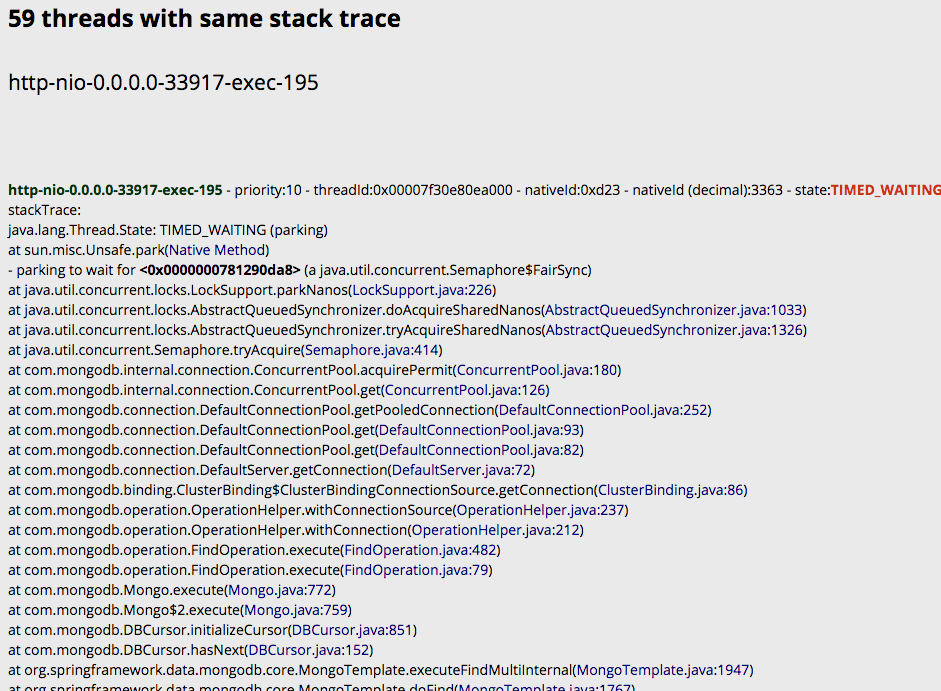


从16：23的jstack可以看出大概有98个MongoDB的读写操作

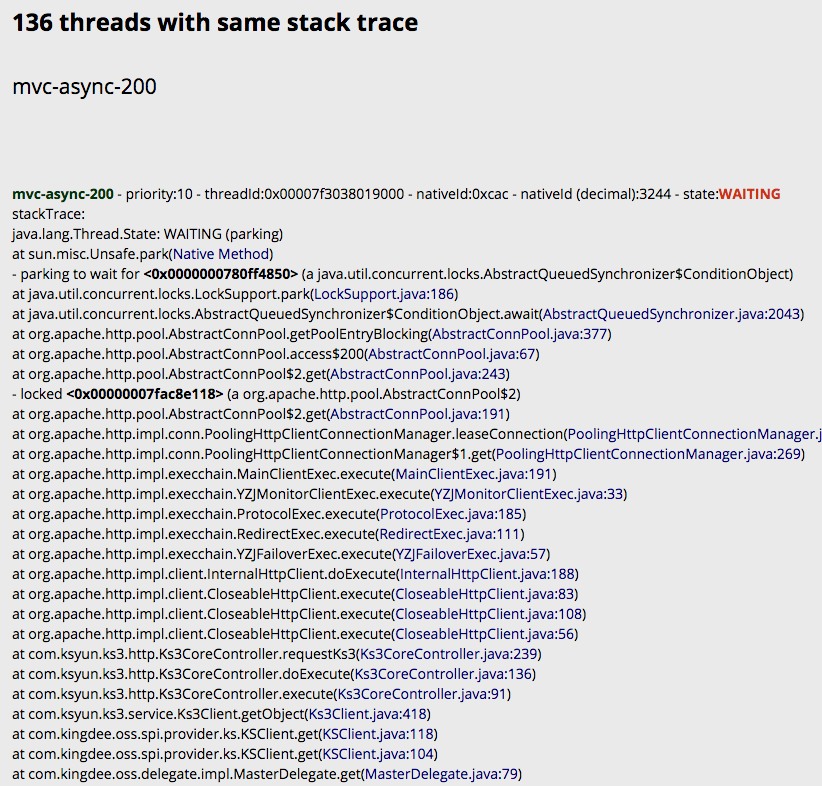


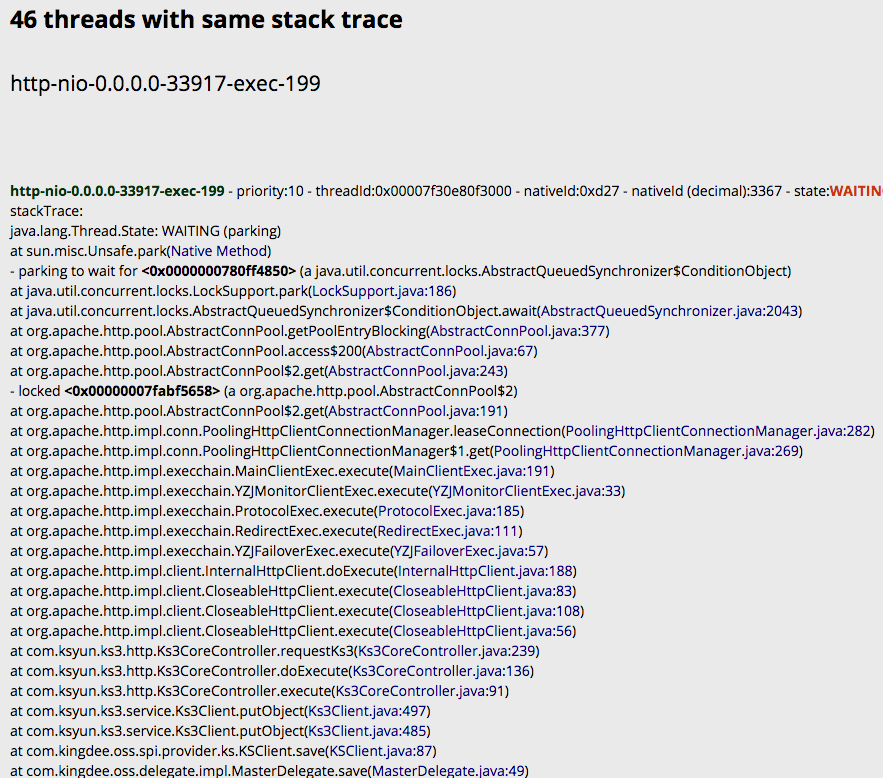
可能是mongodb的读写缓慢。

还有大量线程堵在mongo链接获取。

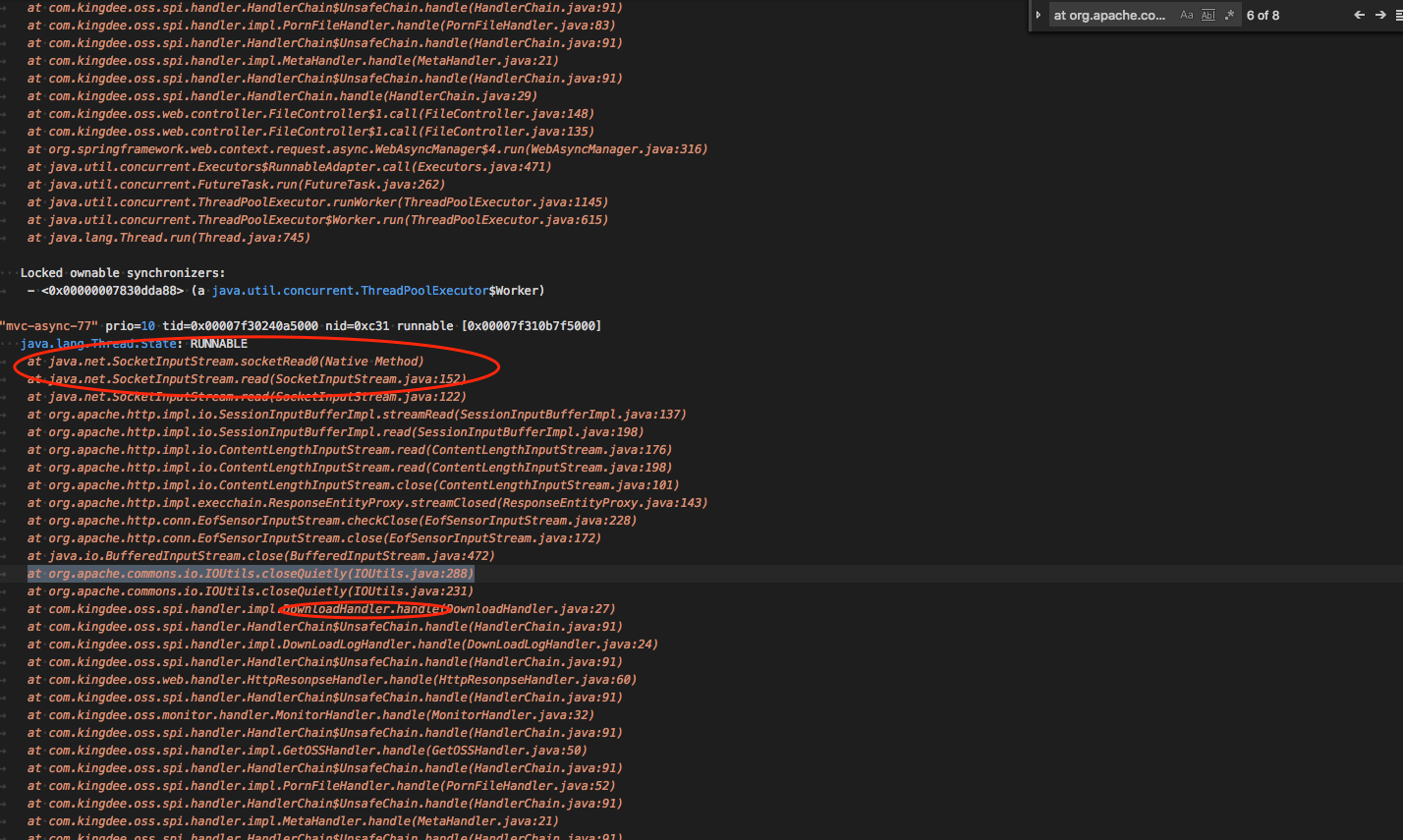


还有大量对ks的读写卡在链接的借用步骤





ks3库链接池配置默认最大为50，可实际只有8个线程在读



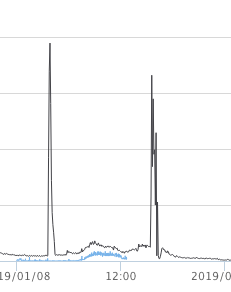
上传的全部卡在链接获取。怀疑参数配置没有生效，依然使用的common-pool的默认最大8的配置。

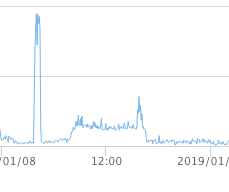
问题有两个：

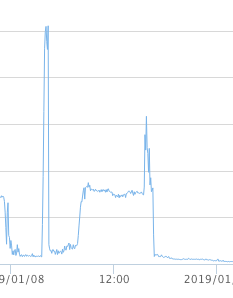
一。Mongo读写缓慢，导致请求堆积，雪崩。

二。对ks3的读写并发大小设置有问题（待进一步确认）。

原因是开发人员在排查文件预览问题时，在文件服务元数据冷库上执行力一条慢查询，用于查找同一md5的文件，该查询没有命中索引，且冷库数据量达到32亿条记录 。该查询正好是在生产的主库执行，导致大量页读取

 <pgpgin>

 <load>

<CPU>



内存使用量增加了35G，



内存读入暴涨



mongo的请求开始出现大量排队。

同时16：38重启数据库后各项基础监控数据开始恢复，同时逐个服务重启后，服务开始恢复正常。

#### 优化策略：

1. 对文件服务冷库进行分片。
2. ks3链接池可以稍微调大一些（默认50）
3. 禁止开发人员自己在生产库上查数据，统一通过工单系统，在DBA对查询语句审批后执行，告知结果。或者新增让开发人员去大数据平台查询，做分析。
4. 完善监控告警系统，增加监控指标及数据收集的精度。有利故障根因分析并作出正确的恢复操作，缩短响应时间。
5. 人员稳定性，项目交接过程优化。