

# 2023. 8. 25电子秤无法传价格故障

故障报障时间：2023-08-25 08:11

故障实际发生时间：2023-08-25 07:56

故障开始处理时间：2023-08-25 08:13

故障恢复时间： 2023-08-25 08:18

故障影响范围：6家店商品价格/资料下秤失败报障, 经排查下秤服务2个分片(总共40个分片, 两个分片上共47家店任务等待执行)任务阻塞.

故障问题描述：门店商品价格/资料传秤失败

故障根因：redis连接卡死hang住，导致job卡死任务阻塞，从而导致商品价格/资料下秤失败。

## 故障处理与分析过程：

### 一、故障处理过程

1、08:11 一运维灭火群报账

## 永辉灭火🔥大队🚒

刘钦涛 @永辉超市 5541

8:11

卢小鹏

多门店报，无法传称，一直显示下传中

刘钦涛 @永辉超市 5541

卢小鹏 8-25 8:11

报障描述：食百，生鲜 电子秤上传不了资料

报障门店：(90L7) 南充蓬安恒丰国际店

报障时间：2023-08-25 08:00:26

报障人：赵大杰

联系电话：18409789612

5541 卢小鹏

报障描述：90G9宿州吾悦生鲜称价格无法下传，帮忙看下什么原因，咨询了顶尖称厂家，不是称的原因，称全部在线，店里也有网络，电脑里价格已生成，称上价格传不进去

报障门店：(90G9) 宿州吾悦广场店

报障时间：2023-08-25 08:10:44

报障人：李燕

🏠 Back to the bottom

后续跟运维要到的实际报障门店, 共有6家门店报障 (90S4, 90L7, 9352, 9650, 90G9, 9540)

8.25电子秤下传															
View only															
Remind															
Insert															
General															
.0															
等线															
11															
B															
I															
U															
S															
A															
田															
三															
+															
=															
田															
Σ															
▽															
Σ															
Data															
...															
Q															
H4	食百, 生鲜 电子秤上传不了资料														
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	工单号	门店编号	门店名称	城市	状态	产品项目	图片	故障文描	故障领域	报障人	报障时间	处理人	处理时间	完成时间	处理意见
2	N169292190A0	合肥市肥	合肥市	已处理	电子秤	https://yw	yhdsos调	现场管理	罗露雲	2023/8/25 07:54	卢小鹏	2023/8/25 07:54	-		3.2192
3	N169292190S4	重庆两江	重庆市	已处理	电子秤	https://yw	yhdsos调	现场管理	罗露雲	2023/8/25 07:56	卢小鹏	2023/8/25 07:56	-		3.1801
4	N169292190L7	南充蓬安	南充市	已处理	电子秤	https://yw	yhdsos调	现场管理	赵大杰	2023/8/25 08:00	卢小鹏	2023/8/25 08:00	-		3.1123
5	N16929219352	淮南市凤	淮南市	处理中	其他	https://yw	yhdsos调	战区	徐志远	2023/8/25 08:05	倪旭	2023/8/25 08:05	-		3.0276
6	N16929219650	铜陵万达	铜陵市	已处理	电子秤	https://yw	yhdsos调	现场管理	金兵	2023/8/25 08:10	卢小鹏	2023/8/25 08:10	-		2.9526
7	N169292190G9	宿州吾悦	宿州市	处理中	其他	https://yw	yhdsos调	战区	李燕	2023/8/25 08:10	倪旭	2023/8/25 08:10	-		2.9406
8	N16929219540	槐房万达	北京市	处理中	其他	https://yw	yhdsos调	槐房店, 战区	刘伟	2023/8/25 08:22	刘威	2023/8/25 08:22	-		2.7403
9															
10															

- 2、08:13 一接到通知进入灭火群进行处理
- 3、一按照之前的排障预案进行处理, 排查ES JOB ,发现两个分片长时间运行中,判断为该分片任务阻塞。
- 4、08:18重启出问题的两台机器

变更内容: 重启容器 yh-sod-scale-deviceadapter-e1576-694f485869-4tb4q

项目空间: 产品-计量中台

变更环境: 生产环境

变更应用: yh-sod-scale-deviceadapter

变更说明: 【负载过高】 重启

变更时间: 2023-08-25 08:18:17

操作人: 刘钦涛#81116320

发布通知 BOT

变更内容: 重启容器 yh-sod-scale-deviceadapter-e1576-694f485869-bwzc6

项目空间: 产品-计量中台

变更环境: 生产环境

变更应用: yh-sod-scale-deviceadapter

变更说明: 【负载过高】 重启

变更时间: 2023-08-25 08:18:34

操作人: 刘钦涛#81116320

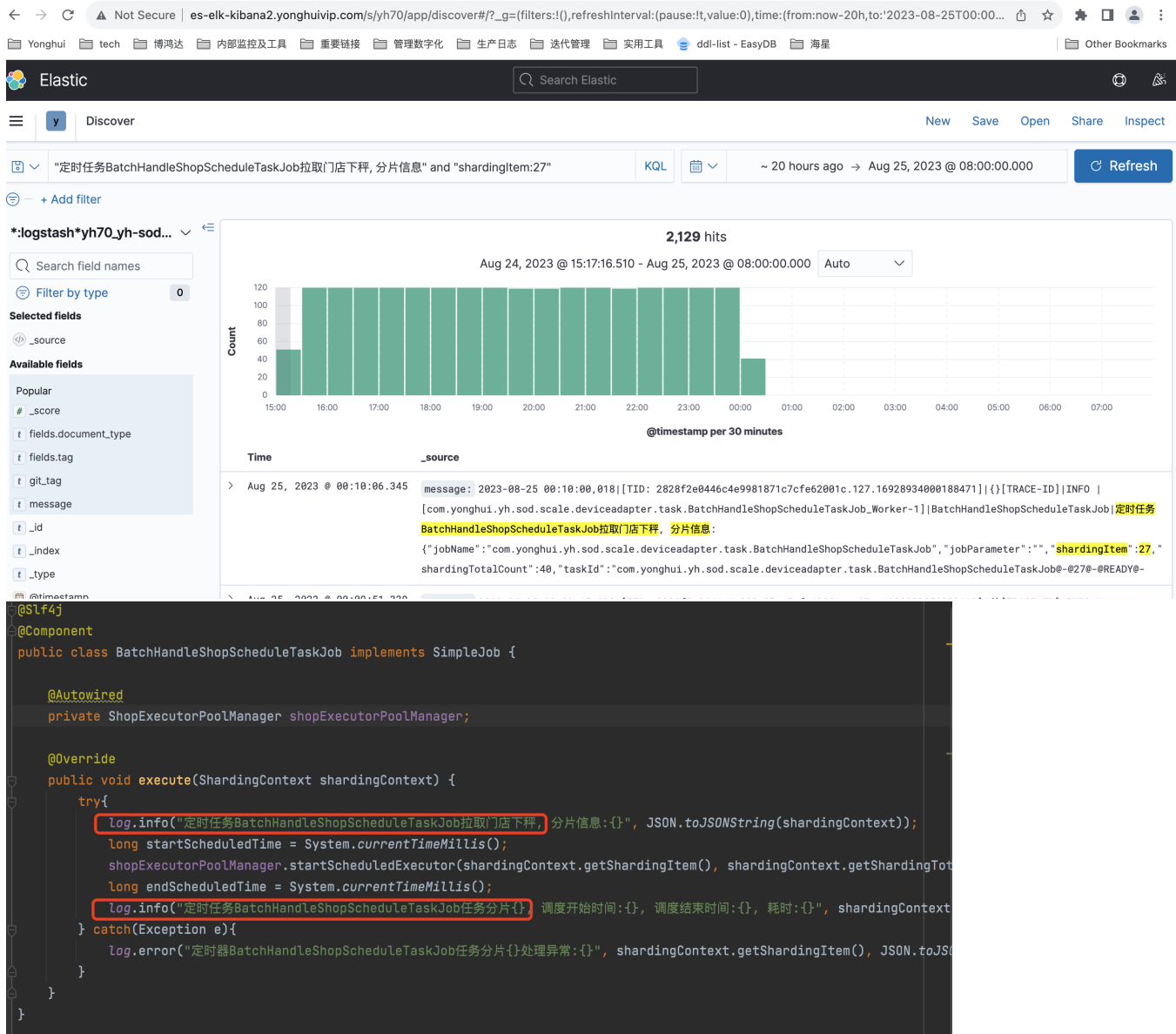
重启后同步灭火群, 运维联系门店进行确认, 研发看数据库任务已经正常处理

根因分析过程:

问题恢复后, 研发进行问题排查, 根据门店找到对应的分片之后,

- 1、发现最后一次调度任务是在00:10 最后一次调度卡住

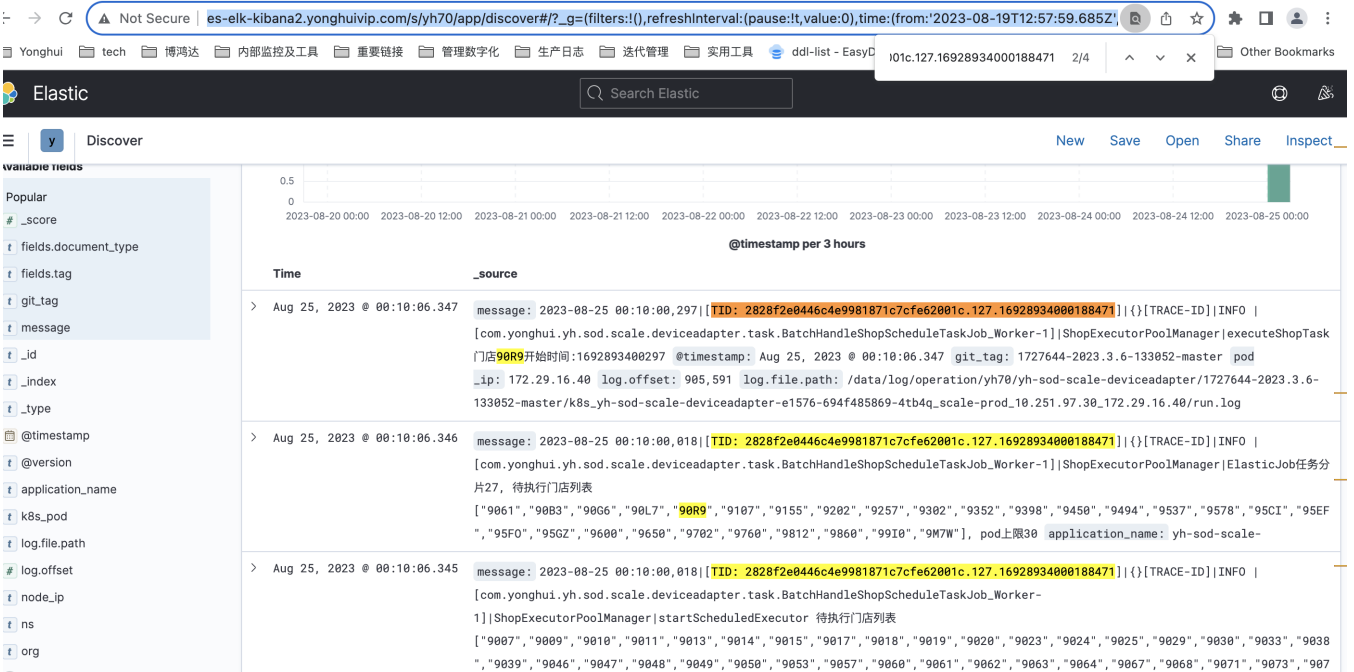
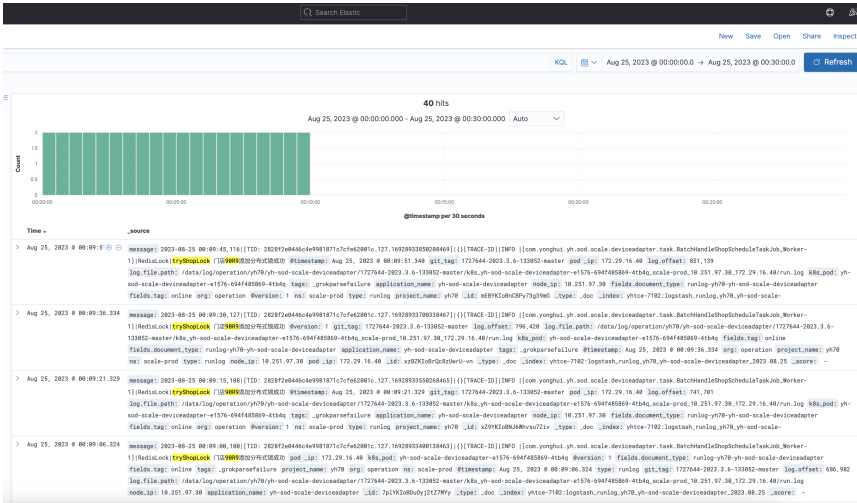
http://es-ek-kibana2.yonghuivip.com/s/yh70/app/discover#/?\_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:now-20h,to:'2023-08-25T00:00:00.000Z'))&a=(columns:!(source),filters:!(),index:'7af6f550-0770-11ed-bd60-4d58239e1af1',interval:auto,query:(language:kuery,query:'%22E5%AE%9A%E6%97%B6%E4%BB%BB%E5%8A%A1BatchHandleShopScheduleTaskJob%E6%8B%89%E5%8F%96%E9%97%A8%E5%BA%97%E4%B8%8B%E7%A7%A4,%20E5%88%86%E7%89%87%E4%BF%A1%E6%81%AF%22%20and%20%22shardingItem:27%22'),sort:!(('@timestamp',desc))



## 2、进一步发现90R9门店的处理流程卡住

可发现, 没有获取锁成功的日志, 只有开始执行的日志

[http://es-elk-kibana2.yonghuivip.com/s/yh70/app/discover#/?\\_g=\(filters:!\(\),refreshInterval:\(pause:!t,value:0\),time:\(from:'2023-08-19T12:57:59.685Z',to:'2023-08-24T16:20:50.531Z'\)\)&\\_a=\(columns:!\(source\),filters:!\(\),index:'7af6f550-0770-11ed-bd60-4d58239e1af1',interval:auto,query:\(language:kuery,query:'%22ID:%202828f2e0446c4e9981871c7cfe62001c.127.16928934000188471%22and%20%2290R9%22'\),sort:!\(timestamp,desc\)\)](http://es-elk-kibana2.yonghuivip.com/s/yh70/app/discover#/?_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:'2023-08-19T12:57:59.685Z',to:'2023-08-24T16:20:50.531Z'))&_a=(columns:!(source),filters:!(),index:'7af6f550-0770-11ed-bd60-4d58239e1af1',interval:auto,query:(language:kuery,query:'%22ID:%202828f2e0446c4e9981871c7cfe62001c.127.16928934000188471%22and%20%2290R9%22'),sort:!(timestamp,desc)))



3、结合日志以及代码, 可以看到与redis交互之后, 至少会有一条与锁相关的日志, 但是上面的日志中没有, 判断为获取redis锁的地方卡住了

```
public void executeShopTask(List<String> shopIdList, Integer shopTaskLimit) {
    long startScheduledTime = System.currentTimeMillis();
    log.info("executeShopTask 批量执行门店开始时间:{}", startScheduledTime);
    // 查找当前门店下的任务
    for (String shopId : shopIdList) {
        long startTime = System.currentTimeMillis();
        log.info("executeShopTask 门店{}开始时间:{}", shopId, startTime);
        try {
            // 判断门店是否达到上限
            if (!validateShopNumsLimit(shopId, shopTaskLimit)) {
                continue;
            }
            // 门店分布式锁, 加锁失败继续下个门店
            if (!redisLock.tryShopLock(shopId)) {
                continue;
            }
            // 查询门店任务
            List<String> scaleIpList = findScaleIpByParam(shopId);
            log.info("executeShopTask 门店{}可执行任务列表{}", shopId, JSONObject.toJSONString(scaleIpList));
            if (CollectionUtils.isEmpty(scaleIpList)) {
                log.info("executeShopTask 门店{}无可执行任务", shopId);
                // 释放门店锁
                redisLock.releaseShopLock(shopId);
                continue;
            }
        }
    }
}
```

4、redisson异常日志(发生在非阻塞的节点上),发现客户端与redis连接异常,发现当时redis连接确实存在异常

Can't update cluster state 异常

http://es-elk-kibana2.yonghuivip.com/s/yh70/app/discover#/?\_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:'2023-08-19T12:57:59.685Z',to:'2023-08-24T16:20:50.531Z'))&\_a=(columns:!(source),filters:!(),index:'7af6f550-0770-11ed-bd60-4d58239e1af1',interval:auto,query:(language:kuery,query:%22redis%22),sort:!(('@timestamp',desc))

博鸿达内部监控及工具重要链接管理数字化生产日志迭代管理实用工具ddi-list - EasyDB海星Other Bookmarks

Search Elastic

erNewSaveOpenShareInspect

f	git_tag	1727644-2023.3.6-133052-master
f	k8s_pod	yh-sod-scale-deviceadapter-e1576-694f485869-cmh29
f	log.file.path	/data/log/operation/yh70/yh-sod-scale-deviceadapter/1727644-2023.3.6-133052-master/k8s_yh-sod-scale-deviceadapter-e1576-694f485869-cmh29_scale-prod_10.251.97.41-172.29.4.40/run.log
f	log.flags	multiline
#	log.offset	160,202,311
f	message	> 2023-08-25 00:10:04,669 [TID: N/A] {}[TRACE-ID] ERROR [redisson-netty-1-7] ClusterConnectionManager Can't execute CLUSTER_NODES with 10.247.44.250/10.247.44.250:6379 org.redisson.client.RedisTimeoutException: Command execution timeout for command: (CLUSTER NODES), command params: [], Redis client: [addr=redis://10.247.44.250:6379] at org.redisson.client.RedisConnection\$1.run(RedisConnection.java:209) [redisson-3.10.4.jar!/:?] at io.netty.util.concurrent.PromiseTask\$RunnableAdapter.call(PromiseTask.java:38) [netty-all-4.1.33.Final.jar!/:4.1.33.Final]
f	node_ip	10.251.97.41
f	ns	scale-prod
f	org	operation
pod	io	172.29.4.40

域名异常

http://es-elk-kibana2.yonghuivip.com/s/yh70/app/discover#/?\_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:'2023-08-24T15:59:00.000Z',to:'2023-08-24T16:12:50.531Z'))&\_a=(columns:!(source),filters:!(),index:'7af6f550-0770-11ed-bd60-4d58239e1af1',interval:auto,query:(language:kuery,query:%22redis.b-scale.mw.yonghui.cn%22),sort:!(('@timestamp',desc))

Not Secure | es-elk-kibana2.yonghuivip.com/s/yh70/app/discover#/?\_g=(filters:!(),refreshInterval:(pause:!t,value:0),time:(from:'2023-08-24T15:59:00.000Z',to:'2023-08-24T16:12:50.531Z'))&\_a=(columns:!(source),filters:!(),index:'7af6f550-0770-11ed-bd60-4d58239e1af1',interval:auto,query:(language:kuery,query:%22redis.b-scale.mw.yonghui.cn%22),sort:!(('@timestamp',desc))

Yonghui | tech | 博鸿达 | 内部监控及工具 | 重要链接 | 管理数字化 | 生产日志 | 迭代管理 | 实用工具 | ddi-list - EasyDB | 海星 | Other Bookmarks

Elastic | Search Elastic

Discover | New | Save | Open | Share | Inspect

redis.b-scale.mw.yonghui.cn | KQL | Aug 24, 2023 @ 23:59:00.0 → Aug 25, 2023 @ 00:12:50.5 | Refresh

+ Add filter

\*:logstash\*yh70\_yh-sod... | Search field names | Filter by type | 0

Selected fields | Available fields | Popular | # \_score | f fields.document\_type | f fields.tag | f git\_tag | f message | f \_id | f \_index | f \_type | @timestamp

2 hits | Aug 24, 2023 @ 23:59:00.000 - Aug 25, 2023 @ 00:12:50.531 | Auto | Count | Time | \_source | > Aug 25, 2023 @ 00:10:02.103 | message: 2023-08-25 00:09:54,228|[TID: N/A]|{}[TRACE-ID]|ERROR|[redisson-netty-1-5]|ClusterConnectionManager|Can't update cluster state io.netty.resolver.dns.DnsResolveContext\$SearchDomainUnknownHostException: Search domain query failed. Original hostname: 'redis.b-scale.mw.yonghui.cn' failed to resolve 'redis.b-scale.mw.yonghui.cn.svc.cluster.local' after 2 queries at io.netty.resolver.dns.DnsResolveContext.finishResolve(DnsResolveContext.java:877) [netty-all-4.1.33.Final.jar!/:4.1.33.Final] at io.netty.resolver.dns.DnsResolveContext.tryToFinishResolve(DnsResolveContext.java:838) [netty-all-4.1.33.Final.jar!/:4.1.33.Final] | > Aug 25, 2023 @ 00:09:55.967 | message: 2023-08-25 00:09:54,228|[TID: N/A]|{}[TRACE-ID]|ERROR|[redisson-netty-1-5]|ClusterConnectionManager|Can't update cluster state io.netty.resolver.dns.DnsResolveContext\$SearchDomainUnknownHostException: Search domain query failed. Original hostname: 'redis.b-scale.mw.yonghui.cn' failed to resolve 'redis.b-scale.mw.yonghui.cn.svc.cluster.local' after 2 queries at io.netty.resolver.dns.DnsResolveContext.finishResolve(DnsResolveContext.java:877) [netty-all-4.1.33.Final.jar!/:4.1.33.Final] at io.netty.resolver.dns.DnsResolveContext.tryToFinishResolve(DnsResolveContext.java:838) [netty-all-4.1.33.Final.jar!/:4.1.33.Final]

## 5、分析redis交互的代码

```
597
598 public <T, R> RFuture<R> writeAsync(byte[] key, Codec codec, RedisCommand<T> command, Object... params) {
599     RPromise<R> mainPromise = this.createPromise();
600     NodeSource source = this.getNodeSource(key);
601     this.async( readOnlyMode: false, source, codec, command, params, mainPromise, attempt: 0, ignoreRedirect: false);
602     return mainPromise;
603 }
```

这里返回mainPromise，它的await方法最终会被调用。

### async中的代码：

DefaultPromise的await调用object.wait

```
public final void wait() throws InterruptedException {
    wait(0);
}
```

```
public Promise<R> await() throws InterruptedException {
    if (this.isDone()) {
        return this;
    } else if (Thread.interrupted()) {
        throw new InterruptedException("Interrupted");
    } else {
        this.checkDeadlock();
        synchronize(this) {
            while(!this.isDone()) {
                this.notify();
            }
        }
        try {
            this.wait();
        } finally {
            this.checkDeadlock();
        }
    }
    return this;
}
```

```
public final native void wait(long timeout) throws InterruptedException;
```

Causes the current thread to wait until another thread invokes the `notify()` method or the `notifyAll()` method for this object, or some other thread interrupts the current thread, or a certain amount of real time has elapsed.

This method is similar to the wait method of one argument, but it allows finer control over the amount of time to wait for a notification before giving up. The amount of real time, measured in nanoseconds, is given by:

有wait() 必定有notify() 或者notifyAll()，而且因为wait() 的对象是this，所以DefaultPromise里面必定有相应的方法来唤醒等待的线程。

```

Decompiled .class file, bytecode version: 50.0 (Java 6)
notifyAll
440 @ private boolean setFailure0(Throwable cause) {
441     return this.setValue0(new DefaultPromise.CauseHolder((Throwable)Obj
442 }
443
444 private boolean setValue0(Object objResult) {
445     if (!RESULT_UPDATER.compareAndSet( obj: this, (Object)null, objResul
446         return false;
447     } else {
448         this.checkNotifyWaiters();
449         return true;
450     }
451 }
452
453 private synchronized void checkNotifyWaiters() {
454     if (this.waiters > 0) {
455         this.notifyAll();
456     }
457 }
458
459 private void incWaiters() {

```

当waiters》0时，会走notifyAll()，但是，

```

public <V, R> void async(boolean readOnlyMode, final NodeSource source, Codec codec, final RedisCommand<V> command, f
    if (mainPromise.isCancelled()) {
        this.free(params);
    } else if (!this.connectionManager.getShutdownLatch().acquire()) {
        this.free(params);
        mainPromise.tryFailure(new RedissonShutdownException("Redisson is shutdown"));
    } else {
        Codec codecToUse = this.getCodec(codec);
        final AsyncDetails<V, R> details = AsyncDetails.acquire();
        final RFuture<RedisConnection> connectionFuture = this.getConnection(readOnlyMode, source, command);
        RPromise<V> attemptPromise = new RedissonPromise();
        details.init(connectionFuture, attemptPromise, readOnlyMode, source, codecToUse, command, params, mainPromise);
        BiConsumer<V, Throwable> mainPromiseListener = accept(t, u) -> {
            if (mainPromise.isCancelled() && connectionFuture.cancel( mayInterruptRunning: false)) {
                CommandAsyncService.log.debug("Connection obtaining canceled for {}", command);
                details.getTimeout().cancel();
                if (details.getAttemptPromise().cancel( mayInterruptRunning: false)) {
                    CommandAsyncService.this.free(params);
                }
            }
        };
        TimerTask retryTimerTask = run(t) -> {
            if (!details.getAttemptPromise().isDone()) {
                if (details.getConnectionFuture().cancel( mayInterruptRunning: false)) {
                    if (details.getException() == null) {
                        details.setException(new RedisTimeoutException("Unable to get connection! Try to increase

```

wait()和notifyAll()分别属于两个不同的对象，导致waiters=0，走不到notifyAll()，从而线程挂起。

与redis相关的类库：

```

yh-starter-redis-0.0.5-RELEASE.jar
redisson-spring-boot-starter-3.10.4.jar
spring-boot-starter-data-redis-2.1.3.RELEASE.jar
spring-data-redis-2.1.5.RELEASE.jar
redisson-3.10.4.jar
redisson-spring-data-21-3.10.4.jar

```

#### 改进措施action:

- 1、增加POD维度的ESJob未触发消息及电话告警逻辑 - @王文斌 0825 已经添加
- 2、ES JOB 异步运行, 保证JOB不阻塞-@王文斌, 2023. 09. 05迭代上线
- 3、增加下秤门店数量/秤数量/任务数量拥堵、失败/成功率、线程数、最大/Tp99耗时 监控、消息和电话告警-@刘钦涛, 2023. 09. 05迭代上线
- 4、下秤上下游链路梳理及技改专项、redis sdk 从redisson改用 jedis-@王文斌 2023. 9. 12

故障责任人：刘钦涛

故障等级: L5