# 问题描述

在nfs server容器挂掉的场景下，nfs client容器被重启会导致kubernetes节点不健康的现象。

# 问题产生

* nfs server挂掉

引起nfs-server挂掉的原因，有许多，一个典型的例子是正常的pod生死（测试中心会反复随机的删除pod和重启容器），伴随pod的正常死亡（网络震荡/节点突然下电等），如果该pod正在对于nfs-server pv对应的iscsi存储卷写入数据，则会引起该文件系统错误，导致nfs-server尝试挂载该iscsi卷失败。

* nfs client挂掉

在nfs-server挂掉的场景下，有可能会引起nfs-client容器的重启操作，引起的原因有许多，例如容器发生OOM或者容器的liveness探针失败，都会触发nfs-client容器的重启。

在nfs server挂掉的场景下，等待并观察apm容器的内存使用率，会发现该容器使用内存一直在不断正常，直到超过设置的4GB内存上线，而被kubelet OOM(out of memory)杀死。

# 问题原因

* docker

docker会为每一个容器提供一把锁，这把锁用于保证对这个容器的增删改查操作是串行的，避免在并发场景下的容器数据不一致。

nfs服务端挂掉会导致docker在启动nfs客户端容器时，stat nfs目录挂住，如果是硬挂载，则会永远挂住，启动容器的方法会永久持有锁，导致该容器后续的增删改查操作都无法进行；如果是软挂载，则会挂住一段时间，后续的正常操作可进行。

docker为每把容器加锁的原因：

<https://github.com/moby/moby/pull/6430>

这个PR是2014年合入的，加入了锁，合入的原因是为了：

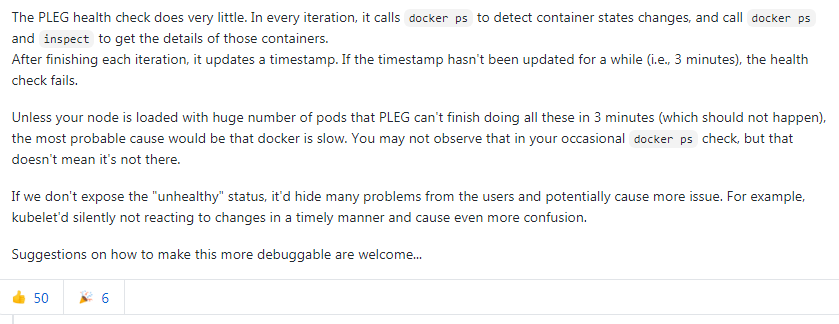
It fixes some race conditions

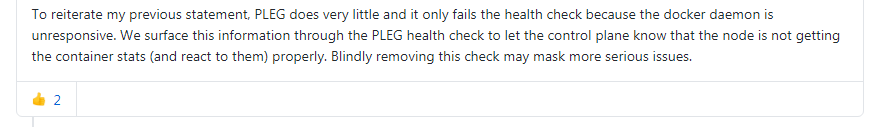
* kubernetes

kubelet PLEG模块会定时每隔一秒访问docker获取所有的容器实时状态，如果在阈值窗口3分钟内，获取任一容器的状态失败，会引起kubelet PLEG模块不健康，导致节点处于NotReady状态。该状态会导致节点不会参与后续的容器调度，并逐步驱逐本节点上的业务容器到其他节点。

nfs服务端挂掉，会导致kubelet PLEG模块获取nfs client容器的状态超时，因为从docker查询nfs客户端容器状态需要获取nfs客户端容器的锁，但是该锁一直被启动容器的方法持有。

kubernete PLEG模块设计原因：





<https://github.com/kubernetes/community/blob/master/contributors/design-proposals/node/pod-lifecycle-event-generator.md>

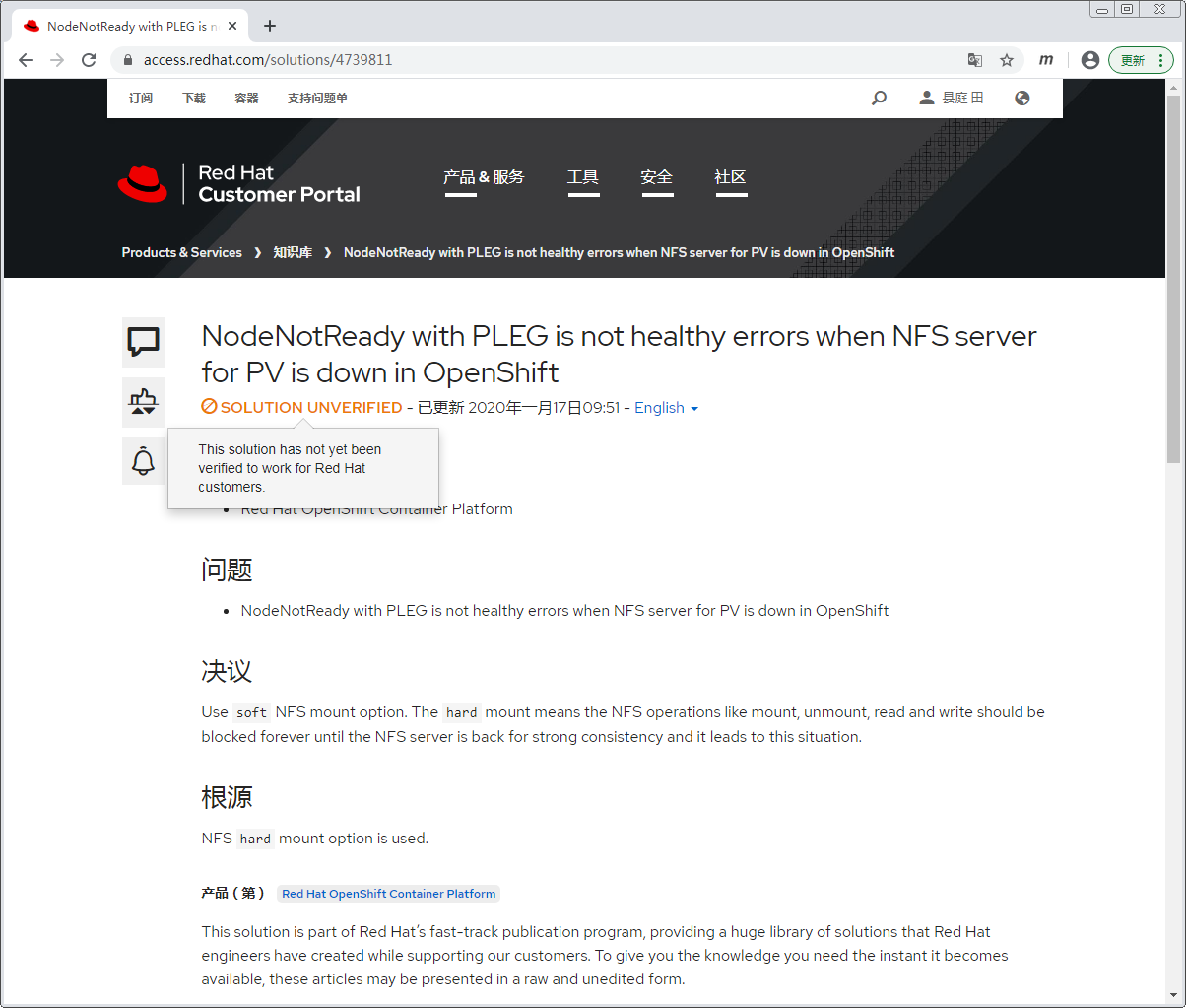
<https://github.com/kubernetes/kubernetes/issues/45419>

PLEG需要实时上报runime和容器的状态，PLEG作者认为在PLEG不健康时，节点应该进入unhealthy状态，尽早的发现错误，上报错误。

# 解决方案

* 使用NFS存储

openshift关于NFS使用的建议：



openshift建议在使用NFS的场景下，不要使用硬挂载选项，否则会导致openshift节点PLEG不健康。

在修改成软挂载后，并不能够完全解决openshift节点PLEG不健康的现象，这个和具体的软挂载时间操作有关，如果软挂载的超时时间和重试次数被限制在PLEG 3分钟健康的阈值内，则不会引起PLEG不健康；反之，则会引起PLEG不健康。

<https://access.redhat.com/solutions/4739811>

<https://access.redhat.com/solutions/3258011>

改了软挂载后，PLEG不健康的现象仍然会出现，关于nfs超时时间和重试次数挂载参数和PLEG不健康的关系：

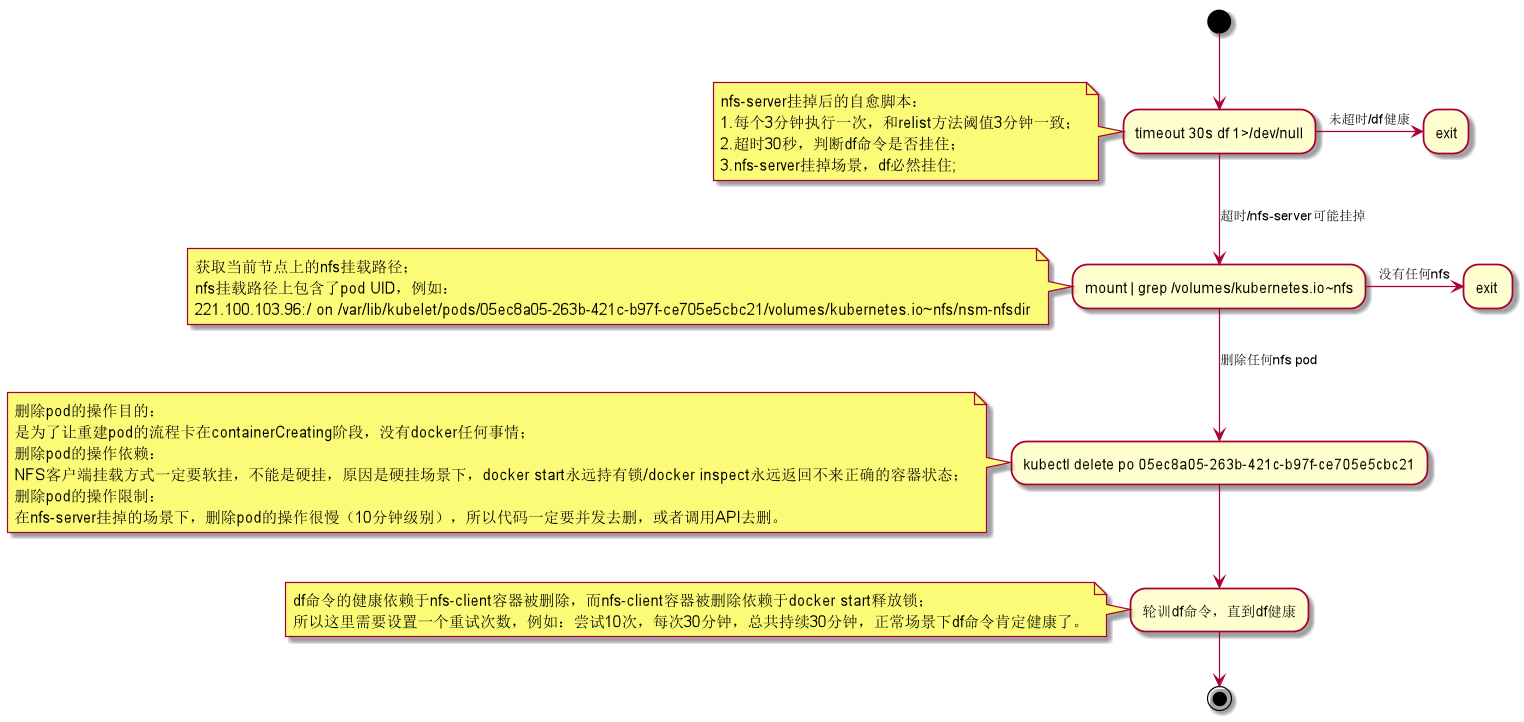
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 挂载参数 | soft  timeo=60sec  retrans=2  nfs客户端数=1 | soft  timeo=30sec  retrans=2  nfs客户端数=1 | soft  timeo=10sec  retrans=2  nfs客户端数=1 | soft  timeo=15sec  retrans=1  nfs客户端数=1 | soft  timeo=15sec  retrans=1  nfs客户端数=5 | soft  timeo=1sec  retrans=1  nfs客户端数=10 |
| stat nfs-path | 3min | 1min32sec | 50sec | 35sec | 35sec | 30sec |
| mkdir nfs-path | 9min | 4min28sec | 2min8sec | 1min48sec | 2min8sec | 52sec |
| Node NotReady | 出现 | 出现 | 不出现 | 不出现 | 出现 | 不出现 |

在单副本的情况下，只需要将软挂载的超时时间设置为15sec，重试次数为1次，即可避免PLEG节点不健康。在多副本的情况下，由于PLEG模块代码串行访问容器获取状态，所以其执行时间是和出问题的副本数呈线性关系，在这种背景下，修改软挂载参数的方法可能无法容忍全部的nfs客户端容器，无法保证节点PLEG健康。当然，如果将超时时间改的非常短，也可容忍多个nfs客户端容器。

注意，docker调用stat和mkdir创建nfs路径，当nfs server挂掉时，其具体的方法执行时间，是和环境中出问题的nfs client客户端成非线性递增关系，如果nfs client客户端出问题的较多，则可能docker层面inspect容器状态卡顿时间会很长。

在调整软挂载超时时间为1秒，重试一次的挂载参数下，至多可以容忍10个nfs客户端容器重启并且不会导致PLEG不健康。

为了防止节点PLEG在软挂载的场景下，出现间歇性不健康的现象，使得不健康的节点状态只出现一次，降低后续的影响，Matrix自身提供了对于这种nfs server端挂掉场景下专用的自愈脚本，脚本的大概逻辑如下：



该脚本会在nfs客户端目录stat超时30秒后，执行删除nfs客户端容器对应pod；正常场景下不会有任何操作；异常场景下仅会删除出问题的nfs客户端，不会误删。

综上，结合软挂载调整时间和自愈脚本的解决方案，达到的最终效果是在nfs server端挂掉，至多会出现一次节点NotReady的情况（如果调整的挂载参数可以容忍出现问题的nfs客户端数量则不会出现节点NotReady情况），在删除nfs客户端对应的pod后，则节点此后不会出现NotReady情况。

* GlusterFS存储

GlusterFS是基于网络的分布式文件系统，具有多副本镜像的容灾能力，并且可以实现单个服务器故障时，不影响客户端访问。故障服务器恢复时候能够自动平衡恢复副本，对挂载GlusterFS的客户端（应用）是透明的容灾，无需用户干预处理。

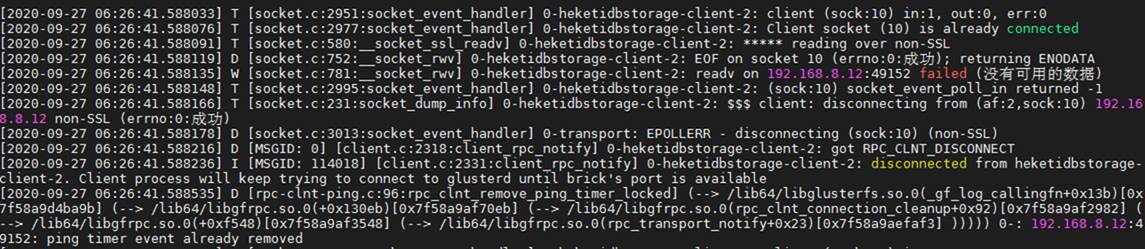
如果GlusterFS后端存储全部挂掉的场景下，并不会出现上述GlusterFS客户端容器异常导致节点NotReady的情况出现。Matrix推荐使用GlusterFS存储。

GlusterFS客户端容器会挂载对应的GlusterFS类型存储卷，在服务端挂掉的场景下，对于该卷的访问直接返回服务端无法连接，不会类似NFS协议那样在stat或者mkdir的系统调用上挂住，客户端容器的状态可以支持获取，不会导致PLEG不健康。

服务端全部故障后：

一旦服务端的故障，客户端会在几秒内感知发现socket连接没有数据，连接服务端的glusterfsd进程失败，日志提示报错，并重新连接，并不会一直保持连接

客户端日志：



一直尝试往各个服务端建立socket连接

