Maintenance Log

Starting to track incidents and how they were resolved.

2019

Date /Time	What happened	How was it resolved
12/16/2019	Disk space errors gfs3	truncated large log files
12/4/2019	Disk space errors loadbal	?? likely truncated large log files
9/9/2019	Disk space errors lma	?? likely truncated large log files / deleted registry cache, possibly purged log/monitoring data
7/11/2019	Pod exceeding restart threshold	Killed pod to reset restart count
6/17/2019	SDSC Maintenance	Brief network outage, then everything automatically came back up
6/13/2019	Disk space errors gfs4	?? likely truncated large log files / deleted registry cache
6/5/2019	Disk space errors node1	Registry was consuming all disk on node1, likely deleted registry cache NRPE daemonset wouldn't run on all nodes. Will run on 7/8 Got it working on node1, then node2 fell off. Manually started nrpe for now.
5/7/2019	SDSC Maintenance	Brief network outage, then everything automatically came back up
4/4/2019	Disk space errors lma	?? likely truncated large log files / deleted registry cache, possibly purged log/monitoring data
3/29/2019	Pod exceeding restart threshold	?? Likely killed pod to reset restart count
3/1/2019	Disk space errors gfs2	Kube registry and GLFS client pods were using ~1.5GB each. MISTAKE: Deleted pods from master to clear old log files. Found that Docker doesn't actually release space from deleted resources until the daemon is restarted. **Required a restart of the Docker daemon on gfs2 to resolve after deleting pods** See https://github.com/moby/moby/issues/21925 In the future, truncating huge log files with the following method is preferred: `echo " " > big-log-file.json`

2018

Date /Time	What happened	How was it resolved

12/31 /2018	Pod exceeding restart threshold	Killed pod to reset restart count
10/27 /2018	Pod exceeding restart threshold	Killed pod to reset restart count
10/26 /2018	Disk space errors gfs2	Same as below on gfs2. We really need to do some cleanup on SDSC and redeploy the beta instance).
9/24/2018	Disk space errors gfs2	Kube registry and GLFS client pods were using ~1.5GB each. Cleared out the offending log files without restarting any containers using `echo " " > big-log-file.json`
9/7/2018 - 9/8/2018	Disk space errors gfs1	Gluster client log file was using 9.6GB. Cleared out the log file without restarting any containers using `echo " " > big-log-file.json`
8/12/2018	Disk space errors lma	Gluster client log file was using way too much space Cleared out the log file without restarting any containers using `echo " " > big-log-file.json`
8/10/2018	Disk space errors gfs4	Gluster client log file was using way too much space. Cleared out the log file without restarting any containers using `echo " " > big-log-file.json`
7/24/2018	load warnings gfs2/node2 /loadbal	Load warnings returned on these same three nodes again, and continued for several hours. This issue is still unresolved, as the load warnings stopped after a time without any obvious manual intervention.
7/2/2018	load warnings loadbal	More unexplained load warnings on loadbal.
6/28/2018	load warnings gfs	More unexplained load warnings on gfs2. Cause is still unknown, but we think this may be related to when users are accessing the NBI data.
6/26/2018	load warnings pod restart warnings NBI data loss	Several Pods went into CrashLoopBackoff as a result of the NBI data being somehow reset. MongoDB reported the size as 500MB, instead of the expected ~20GB. NBI was scaled down and the data was restored (I think?)
6/18 /2018 - 6 /22/2018	load warnings gfs2/node2 /loadbal	Still unexplained - load warnings started popping up on these three nodes and continued for several hours. This issue is still unresolved, as the load warnings stopped after a time without any obvious manual intervention.
6/18/2018	SSH brute force attempts all nodes	Noticed a lot of brute force attempts on many of our nodes. Only allowing a subset of NCSA/TACC/SDSC public IPs for now, and my home IP when remote access is needed.
6/14/2018	Disk space errors gfs3	The registry cache was using ~37GB Couldn't exec into cache as below, due to OutOfDisk default docker-cache-gnc8m 0/1 OutOfDisk 0 345d default docker-cache-q1jh2 1/1 Running 0 17m Since the pod had already been moved elsewhere, just deleted it. However, the daemonset wouldn't create the pod on gfs3 unless I edited the spec. Added a simple label (other; test) and the pod appeared.
4/29/2018	gfs2 disk space warnings	Same problem as 4/23/2018 and 2/12/2018, except on GFS2. On nodes where we did not initially plan to execute user services, we did not mount /var/lib/docker. Hopefully in the coming weeks we will be able to reprovision the Workbench Beta to reset the clock on these warnings.
4/23/2018	LMA disk space warnings	Same thing as 2/12/2018... I deleted the jupyter-nbi Docker image from that node (again) to clear up some space. We should probably consider/discuss removing the "compute" node label from this node to prevent it from happening again.

4/10/2018	SSL handshake errors	Nagios NRPE container disappeared from only node2 Performing a "kubectl apply -f ~/nagios-nrpe.ds.yaml" brought it back on Also cleared out some space on node2's /var/lib/docker (it was at 94%) by deleting /var/lib/docker/tmp and restarting the docker daemon
2/12/2018	LMA disk space warnings	LMA node on public beta does not appear to have a **/var/lib/docker** mount.. this would be fine, except that the node also had "ndslabs-role-compute: true" set, so client pods had been scheduled there. This included one instance each of NBI and MDF Forge, each of which have huge images (~4GB) with NBI also having a larger-than-average docker overlay folder. Short term: I have temporarily removed the compute label from LMA and deleted the MDF Forge pod and image - the NBI instance is Akshay's, so I will leave it running to avoid interrupting their work. Long term: Once the user services are gone from this node (e.g. timeout), we can stop the docker daemon on LMA and remount /var/lib/docker as a bind-mount from /media/storage, as is standard on the other nodes.
1/19/2018	Disk space warnings gfs3	The registry cache was using ~34GB disk. kubectl exec -it regsitry sh wget localhost:5001/v2/_catalog -O - (lists images in cache) cd /var/lib/registry/docker/registry/v2 find something that can be removed (e.g., repositories/craigwillis/apiserver) rm -r repositories/craigwillis/apiserver /bin/registry garbage-collect /etc/docker/registry/config.yml Deletes cached blobs
1/14/2018	gfs4 load warnings	Ongoing load warning on gfs4. Noticed gfs2 brick not connected. Restarted gfs2 gluster server. Rebooted gfs4 node. Ran gluster volume heal global info gluster volume heal global to heal files
1/8/2018	transport connection errors	Started receiving alerts about exceeded pod restart thresholds for two mongo containers. Noticed I/O errors in mongo logs. Exec'd into Gluster server and noted that two bricks (node1, node2) were offline. Restarted both pods, one at a time.

2017

Date /Time	What happened	How was it resolved
11/16 /2017	net/http request failures pulling images	Ongoing issue with ETK instance on Nebula, large image pulls are exceeding Kubernetes timeouts. We've decided to migrate the ETK instance to the new Jetstream allocation until the Nebula filesystem problems are resolved.
10/31 /2017 8:30am	loadbal out of disk	I was able to clear out ~500MB of space by deleting Dead/Exited containers, but this problem is still ongoing. Craig is experimenting with enabling docker logs rotation on this node, to prevent us from needing to check on it once every two weeks.
10/24 /2017 10: 30am	node1 acting sluggish almost out of space on /var/lib /docker (88 %)	Ran the following on node1 to remove any images that no longer have valid tags (i.e. images that have been updated that have stale references cached that will never be started.. mysql releasing a new tag for 5.7, rebuilding cloud9, etc): `docker images | grep none && docker rmi $(docker images | grep none | awk '{print $3}')`

10/20/2017 ~4:30pm	lma / node1 pods hung in **Terminating** / **ContainerCreating** state	Node1 hit a weird zombie problem where Kenton's mongo instance would not terminate. Rebooting node1 allowed the shutdowns to complete properly, but this caused some odd behavior in the MTU settings... LMA node was running ElasticSearch / kibana, which filled up /media/storage with 19GB of log data, with 2 replicas took up 38GB of the 40GB storage drive. Unable to kill ElasticSearch due to hung/zombie pods. Looking into it a bit further, zombies seemed to be caused by MTU mismatch between **docker0** (1500 == incorrect) and **flannel.1** (1408 == correct) Further inspection revealed that **/etc/systemd/system/docker.service.d/10-docker0.conf** had specified 1454 as the MTU (also incorrect). Changing this to the correct value of 1408 and running **docker network inspect bridge** / **ifconfig** now shows the correct docker0 MTU. Resetting the MTU allowed the hung pods to finish shutting down/starting up, and I was then able to shut down the running elasticsearch / kibana to automatically clear out the storage drive.

10/13/2017
 	
~2:30pm
	master kube services dead	NAGIOS was complaining: "workbench-master1/Kubernetes Pods is UNKNOWN: CRITICAL: Get http://localhost:8080/api/v1/pods: dial tcp 127.0.0.1:8080: getsockopt: connection refused"
		
		Fixed by running the following:
		
		`sudo systemctl start kube-apiserver kube-scheduler kube-controller-manager`
		
		`# These last two were probably not necessary, but just in case...`
		`sudo systemctl enable kube-apiserver kube-scheduler kube-controller-manager`
		`sudo systemctl unmask kube-apiserver kube-scheduler kube-controller-manager`

9/24/2017 ~1pm	loadbal out of disk	NAGIOS alerted that node was nearly out of disk space (again). Mike restarted the ilb pod to clear out the log file. This did not appear to alleviate the symptom, so he also restarted the node with a **sudo reboot**. NOTE: This reboot reset the MTU settings on the node. Please remember to verify MTU settings after reboot
8/9/2017 ~7pm	loadbal out of disk	NAGIOS alerted that node was nearly out of disk space (again). Mike restarted the loadbalancer node with a **sudo reboot**
8/8/2017 ~3am	loadbal out of disk	NAGIOS alerted that node was nearly out of disk space. Craig restarted the ilb pod to clear out the huge 9.5GB log file.

7/22/2017	Single Pod restart threshold surpassed	NAGIOS started complaining shortly after 7pm: "workbench-master1/Kubernetes Pods is WARNING: 1 pods exceeding WARNING restart threshold."
		
		A Fedora Commons pod had restarted a sixth time (due to OOMKilled), which started triggering these warnings.
		
		Solution was to delete the pod in question to reset the restart count.
		
		`# List pods Sorted by Restart Count`
		`$ kubectl get pods --all-namespaces --sort-by='.status.containerStatuses[0].restartCount'`

7/4 /2017	gfs4 out of disk	`/media/storage ran out of disk, due to the docker cache pod filling up the disk... pod had already been recreated so I looked up the uuid of the broken pod, deleted it, and SSH'd into gfs4 to delete its folder from`
		
		`core@workbench-master1 ~ $ kubectl get pods -o wide`
		`NAME                          READY   STATUS      RESTARTS   AGE    IP            NODE`
		`default-http-backend-zjhdb    1/1     Running     1          23d    10.100.35.5   loadbal`
		`docker-cache-fwkd6            0/1     OutOfDisk   0          130d   <none>        gfs4`
		`docker-cache-gnc8m            1/1     Running     0          4h     10.100.33.5   gfs3`
		
		`core@workbench-master1 ~ $ kubectl get pod -o yaml docker-cache-fwkd6 | grep uid`
		` uid: bf5284e8-fa16-11e6-9d8b-fa163e19eb19`
		
		`core@workbench-gfs4 ~ $ sudo su`
		`workbench-gfs4 core # rm -rf /var/lib/kubelet/pods/bf535ef3-fa16-11e6-9d8b-fa163e19eb19`
		
		---- 	
		
		nagios pod was missing on gfs4 after this, ~~so I had to restart the whole daemonset~~ but thankfully a kubectl apply on the nagios YAML recreated the missing pod without touching the working ones
		
		`core@workbench-master1 ~ $ kubectl get ds nagios-nrpe --namespace=kube-system`
		`NAME          DESIRED   CURRENT   READY   NODE-SELECTOR   AGE`
		`nagios-nrpe   7         7         7       <none>          130d`
		
		`core@workbench-master1 ~ $ kubectl apply -f nagios-nrpe-ds.yaml`
		`daemonset "nagios-nrpe" configured`
		
		`core@workbench-master1 ~ $ kubectl get ds nagios-nrpe --namespace=kube-system`
		`NAME          DESIRED   CURRENT   READY   NODE-SELECTOR   AGE`
		`nagios-nrpe   8         8         8       <none>          130d`

2/9 /2017	Multiple instances	Multiple instance I/O errors across projects, apparently due to Gluster outage on Nebula. Problem first detected at 3AM, reported at 6AM. No updates as of 10:30AM.
1/29 /2017	node1	-bash: /usr/bin/wc: Input/output error Gluster problems on Nebula
1/25 /2017	GFS nodes	Multiple incidents of GFS server pods not responding during healthz. In all cases, one or more glfs-server pods will not respond to exec. SSH to GFS node is find, but docker is unresponsive (docker ps hangs). journalctl shows errors related to registry cache Jan 28 11:10:23 workbench-gfs4.os.ncsa.edu dockerd[26174]: time="2017-01-28T11:10:23.365820903-06:00" level=warning msg="Error getting v2 registry: Get http://localhost:5001/v2/: read tcp 127.0.0.1:36906->127.0.0.1:5001: read: connection reset by peer" Jan 28 11:10:23 workbench-gfs4.os.ncsa.edu dockerd[26174]: time="2017-01-28T11:10:23.365843976-06:00" level=error msg="Attempting next endpoint for pull after error: Get http://localhost:5001/v2/: read tcp 127.0.0.1:36906->127.0.0.1:5001: read: connection reset by peer" Generally, restarting docker daemon temporarily resolves problem.
1/20 /2017	Node1 /Node3 unavailable	Nodes 1 and 3 where not accessible via SSH from the Nagios instances. Node3 was totally unaccessible – Horizon console indicated OOM. Hard reboot succeeded, but CoreOS upgraded to 1235, introducing the flannel error. Copied the flannel config to /run/flannel and restarted. Node3 was accessible, but docker was down. Restarting docker failed until /var/lib/docker was deleted. Also upgraded to 1235, requiring the flannel change.
	OPS node read-only	OPS node is currently in read-only state (same old Nebula problem). Should be resolved by reboot when needed.
	Master Kubelet down	Master Kubelet died due to etcd memory error (known issue). Rebooted, CoreOS upgrade required flannel fix.
1/12 /2017	Loadbalancer sluggish	workbench-loadbal has been sluggish, slow response times resulting in numerous false positive nagios alerts. At some point this afternoon, it was unresponsive. Hard reboot via Horizon took >30 minutes for CoreOS1122 (which takes ~30 seconds on a normal day). Login was slow after reboot, services never fully revived. David suggests that this is a storage problem, but Nebula team can find no apparent cause. Starting standalone CoreOS instances works without error. Tried two different approaches: 1. shutdown -h of the instance and restart to see if hypervisor moves somewhere more friendly. 2. create a snapshot of another node (lma) and use this to create a new instance from it. After boot, edit /etc/kubernetes/kubelet change KUBELET_HOSTNAME from lma to loabal, systemctl restart kubelet. After this, kubectl get nodes showed loadbal in ready state with correct label. Disassociated the IP, associated with new instance. Shutdown bad instance.
1/4 /2017	GFS4 not accessible.	Resolved 1/4 by Nebula team – continued problem with Gluster server.

2016

Date /Time	What happened	How was it resolved
12/26 /2016	GFS1 not accessible. Rebooting via Nebula put node in error state	Resolved on 1/3 by Nebula team – apparent problem with Gluster server. Node was able to restart.
11/8 /2016	API server not accessible – all Kubernetes services down on workbench-master1	It again appears that etcd2 went down, probably due to memory problems. Rebooted the node.
11/4 /2016	NAGIOS error for labstest-lma	Same as above. Nebula team resolved the glusterfs issue. Did not have permission to issue the reset state command.
11/3 /2016	NAGIOS errors "could not complete SSH Handshake" node6	Looked at node6 console via Nebula. Appears to be OOM problem (maybe old swap issue?). kubectl get nodes says all nodes except node6 are ready. Node is totally inaccessible. Tried soft reboot via Horizon, but node was then in error state. Spoke with Nebula group, this was related to the error from Monday. They resolved the underlying problem, but I still wasn't able to start the instance. Using cli: nova show <instance> nova reset-state --active <instance> nova start instance Did the trick
10/31 /2016 ~8am	NAGIOS errors on gfs2, node1, node3	Attempted to reboot nodes, but encountered error: > "Error: Failed to perform requested operation on instance "workbench- > gfs2", the instance has an error status: Please try again later > [Error: cannot write data to file '/etc/libvirt/qemu/instance- > 00002cf1.xml.new': No space left on device]." Emailed Nebula group – apparently a problem with their glusterfs. Resolved at 10:30 AM