



KubeCon

North America 2017

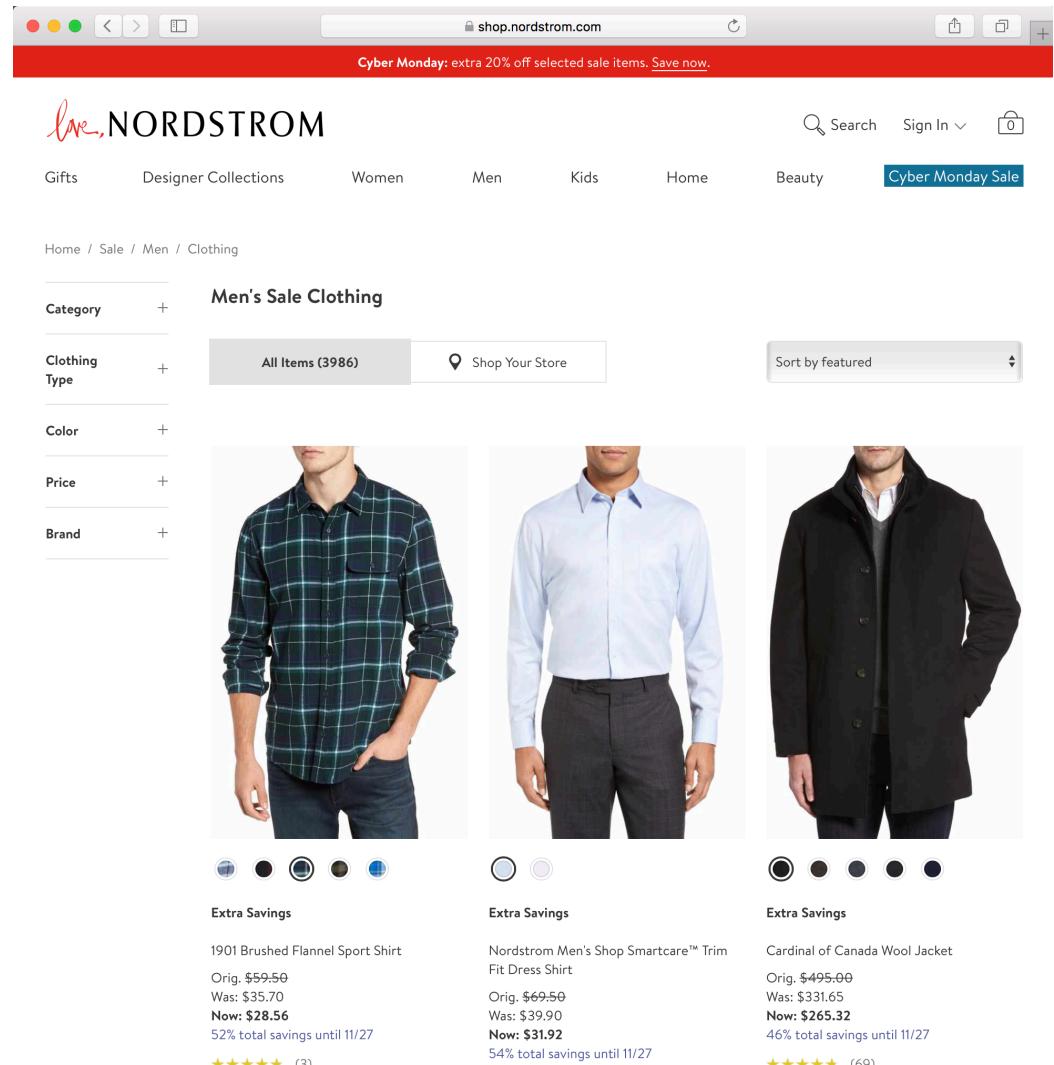
101 Ways to Crash Your Cluster

Marius Grigoriu, Sr. Technology Manager, Nordstrom
Emmanuel Gomez, Principal Engineer, Nordstrom

Kubernetes at Nordstrom

μServices

- Reviews
- Giftcard
- Purchase Orders
- Authentication
- Personalization
- And more



Kubernetes at Nordstrom

Dev Tools

- Issue tracking
- Build runners
- Log aggregation
- Telemetry aggregation
- Alerting





The Tale of the Unresponsive Node

Once upon a time, we were alerted to a few nodes going NotReady.

So we described the node to find out what was being reported...

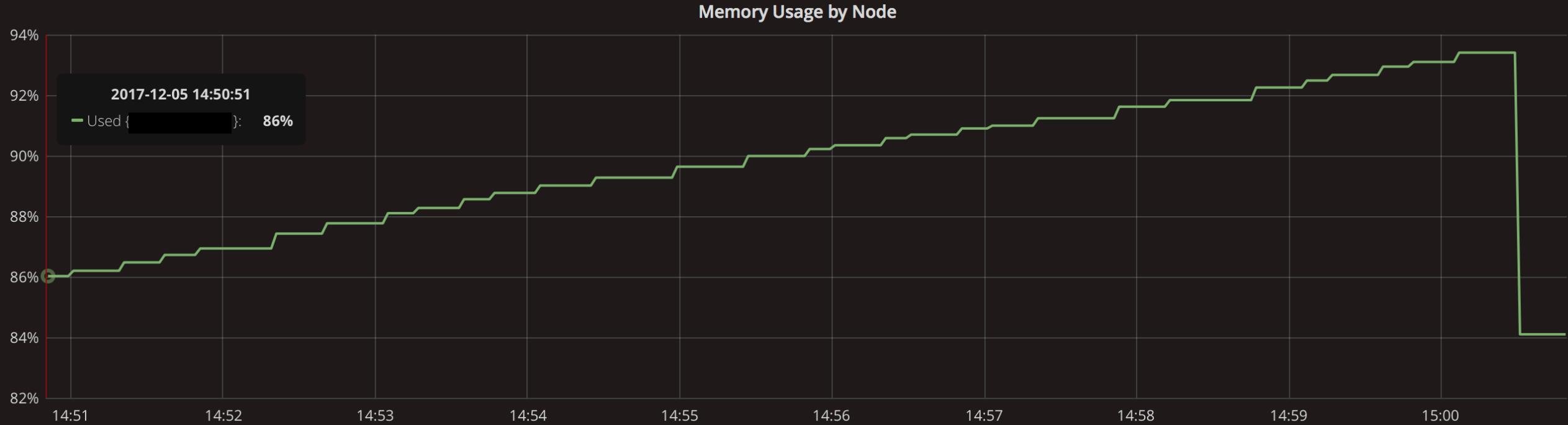
Prometheus APP 12:59 AM

[NodeNotReadyForTooLongJustOne:1, hydrogen] Node ip: [REDACTED] us-west-2.compute.internal has been in a NOT READY state for more than 9m, that's generally not awesome.

Node ip: [REDACTED] us-west-2.compute.internal has been in a NOT READY state for more than 9m, this should be investigated.

kubelet stopped posting status

Looking into the past



The Tale of the Unresponsive Node

```
$ dmesg -HT  
[Tue Nov 14 04:11:38 2017] stress: page allocation stalls for 10047ms,  
order:0, mode:0x14280ca  
[Tue Nov 14 04:11:24 2017] Out of memory: Kill process 40884 (sh)  
score 999 or sacrifice child  
[Tue Nov 14 04:11:24 2017] Killed process 91984 (dd) total-vm:4420kB,  
anon-rss:76kB, file-rss:0kB, shmem-rss:0kB  
[Tue Nov 14 04:11:24 2017] oom_reaper: reaped process 91984 (dd), now  
anon-rss:0kB, file-rss:0kB, shmem-rss:0kB
```



NotReady Node Troubleshooting Steps

1. Run: `kubectl describe node`
2. “Kubelet stopped posting node status”
3. Look for signs of high resource utilization
4. Search through kernel messages (`dmesg`) if suspecting OOM kills

Set your kubelet flags correctly

- Set eviction thresholds
 - <https://kubernetes.io/docs/tasks/administer-cluster/out-of-resource/>
 - Evict early
- Reserve enough resources for kubelet and system daemons
 - <https://kubernetes.io/docs/tasks/administer-cluster/reserve-compute-resources/>
 - --kube-reserved and --kube-reserved-cgroup
 - --system-reserved and --system-reserved-cgroup



Lessons Learned

NAME	STATUS	ROLES	AGE	VERSION
ip-12.us-west-2.compute.internal	NotReady	<none>	11d	v1.7.6+coreos.0
ip-18.us-west-2.compute.internal	NotReady	<none>	69d	v1.7.6+coreos.0
ip-50.us-west-2.compute.internal	NotReady	<none>	74d	v1.7.6+coreos.0
ip-76.us-west-2.compute.internal	NotReady	<none>	13d	v1.7.6+coreos.0
ip-96.us-west-2.compute.internal	NotReady	<none>	27d	v1.7.6+coreos.0
ip-21.us-west-2.compute.internal	NotReady	<none>	42d	v1.7.6+coreos.0
ip-7.us-west-2.compute.internal	NotReady	<none>	11d	v1.7.6+coreos.0
ip-9.us-west-2.compute.internal	NotReady	<none>	10d	v1.7.6+coreos.0
ip-.us-west-2.compute.internal	NotReady	<none>	12d	v1.7.6+coreos.0
ip-0.us-west-2.compute.internal	NotReady	<none>	74d	v1.7.6+coreos.0
ip-27.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-47.us-west-2.compute.internal	NotReady	<none>	74d	v1.7.6+coreos.0
ip-67.us-west-2.compute.internal	NotReady	<none>	74d	v1.7.6+coreos.0
ip-94.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-33.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-5.us-west-2.compute.internal	NotReady	<none>	61d	v1.7.6+coreos.0
ip-.us-west-2.compute.internal	NotReady	<none>	11d	v1.7.6+coreos.0
ip-2.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-39.us-west-2.compute.internal	NotReady	<none>	20d	v1.7.6+coreos.0
ip-9.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-24.us-west-2.compute.internal	NotReady	<none>	74d	v1.7.6+coreos.0
ip-49.us-west-2.compute.internal	NotReady	<none>	11d	v1.7.6+coreos.0
ip-5.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-22.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-3.us-west-2.compute.internal	NotReady	<none>	74d	v1.7.6+coreos.0
ip-58.us-west-2.compute.internal	NotReady	<none>	11d	v1.7.6+coreos.0
ip-71.us-west-2.compute.internal	NotReady	<none>	39d	v1.7.6+coreos.0
ip-85.us-west-2.compute.internal	NotReady	<none>	61d	v1.7.6+coreos.0
ip-1.us-west-2.compute.internal	NotReady	<none>	10d	v1.7.6+coreos.0
ip-2.us-west-2.compute.internal	NotReady	<none>	12d	v1.7.6+coreos.0
ip-3.us-west-2.compute.internal	NotReady	<none>	13d	v1.7.6+coreos.0
ip-7.us-west-2.compute.internal	NotReady	<none>	74d	v1.7.6+coreos.0
ip-1.us-west-2.compute.internal	NotReady	<none>	55d	v1.7.6+coreos.0
ip-25.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-27.us-west-2.compute.internal	NotReady	<none>	75d	v1.7.6+coreos.0
ip-42.us-west-2.compute.internal	NotReady	<none>	20d	v1.7.6+coreos.0
ip-4.us-west-2.compute.internal	NotReady	<none>	11d	v1.7.6+coreos.0
ip-7.us-west-2.compute.internal	NotReady	<none>	12d	v1.7.6+coreos.0

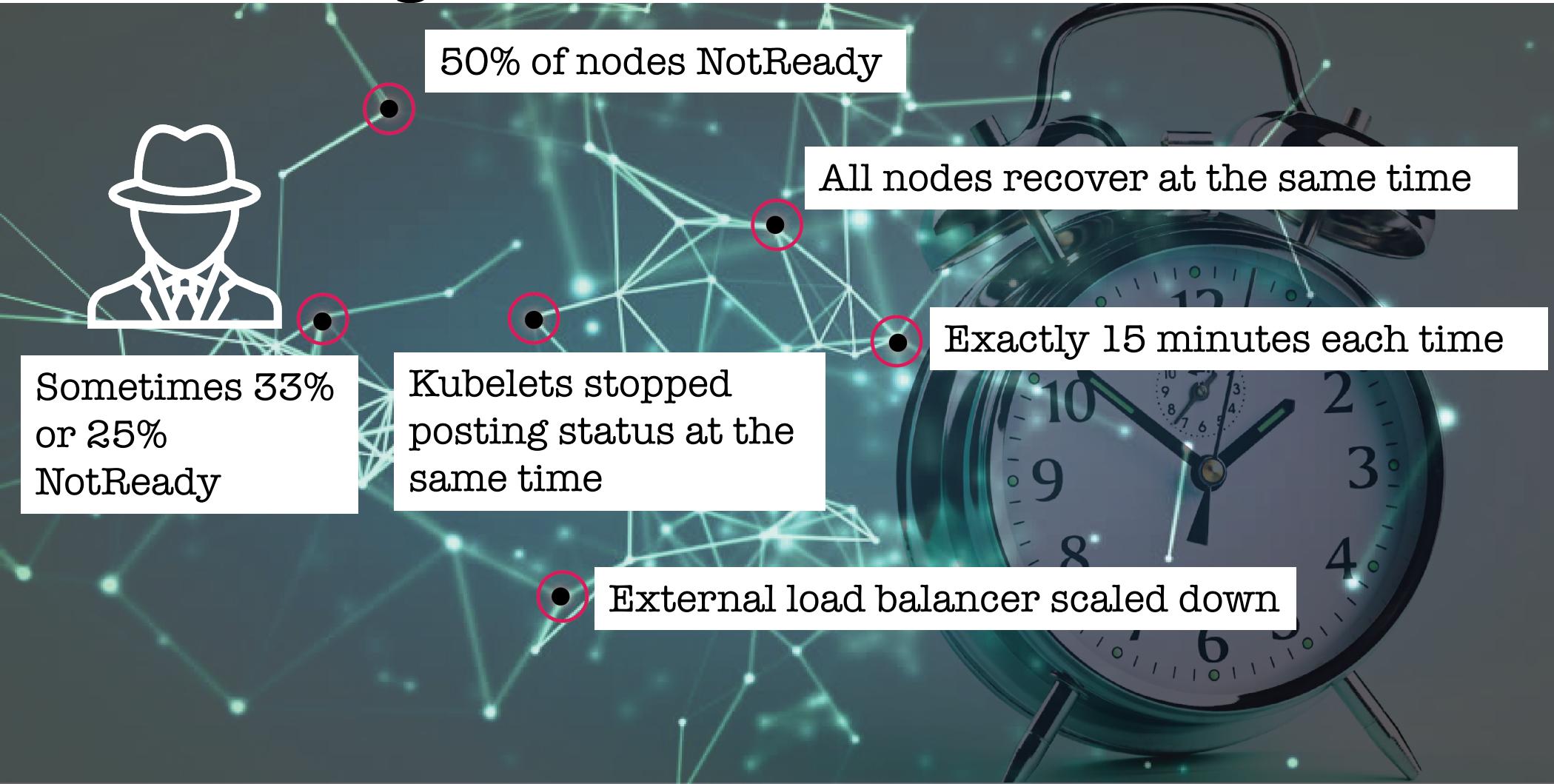


DON'T PANIC

NotReady Storm Checklist

1. Run: `kubectl describe node`
2. “kubelet stopped posting node status”
3. Look for signs of high resource utilization
4. Is there a networking issue?
5. kubelets messed up?
6. apiserver messed up?
7. Oh look, everything’s OK now

Following the trail of clues



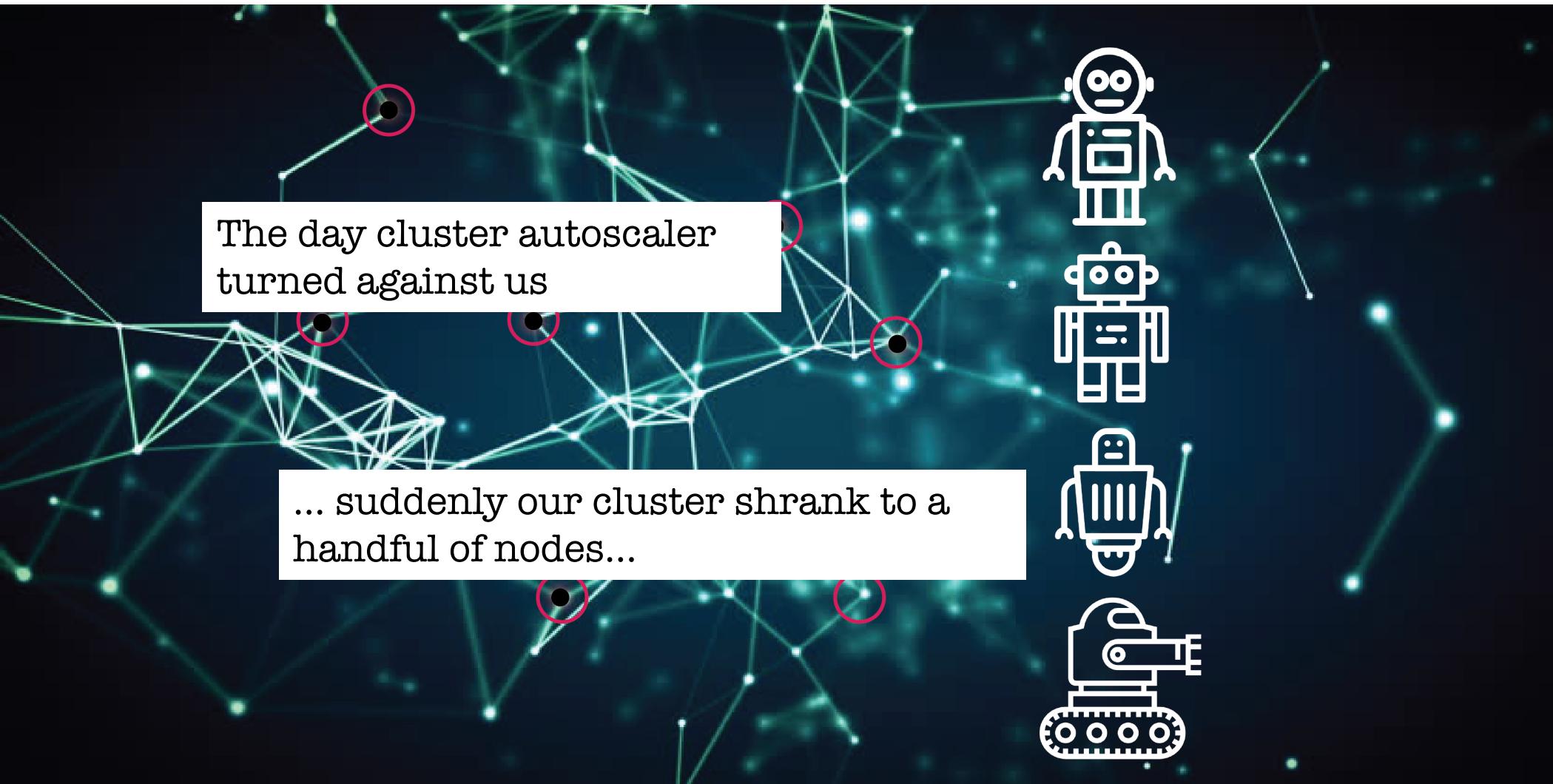
Fixes for the Node NotReady Storm

- Caused by lack of timeout or heartbeat kubelet->apiserver
 - <https://github.com/kubernetes/kubernetes/issues/48638>
- Switch from Elastic Load Balancer to Network Load Balancer
 - “NLB handles connections with built-in fault tolerance, and can handle connections that are open for months or years”
- Fixed in 1.8 (backported to 1.7.8)
 - <https://github.com/kubernetes/kubernetes/pull/52176>



Lessons Learned

The Day The Autoscaler Robots Turned Against Us



All the nodes went away checklist

1. kubectl get nodes shows only a handful of nodes
2. Look at ASG logs
3. Look at cluster autoscaler logs
4. Find utilization of 0.0

The [Cluster Autoscaler] Robots

- We have not been able to determine true root cause
 - Diagnostic data aged out
 - Open-ended work of diagnosing yielded to pressure to move on
- Mea culpa—we should have:
 - Durably captured our diagnostic data (logs, metrics, etc)
 - Promptly opened an upstream issue



Lessons Learned

The [Cluster Autoscaler] Robots

- We worked around it
 - Extended ‘smoothing function’ (min scale down) to 40 minutes
- Better still (but not yet implemented)
 - Alert when planning to scale down too low
 - We don’t have a good way to alert for what we want
 - Need a metric on number of nodes that **will be** scaled in, not number of nodes that are unneeded



Lessons Learned

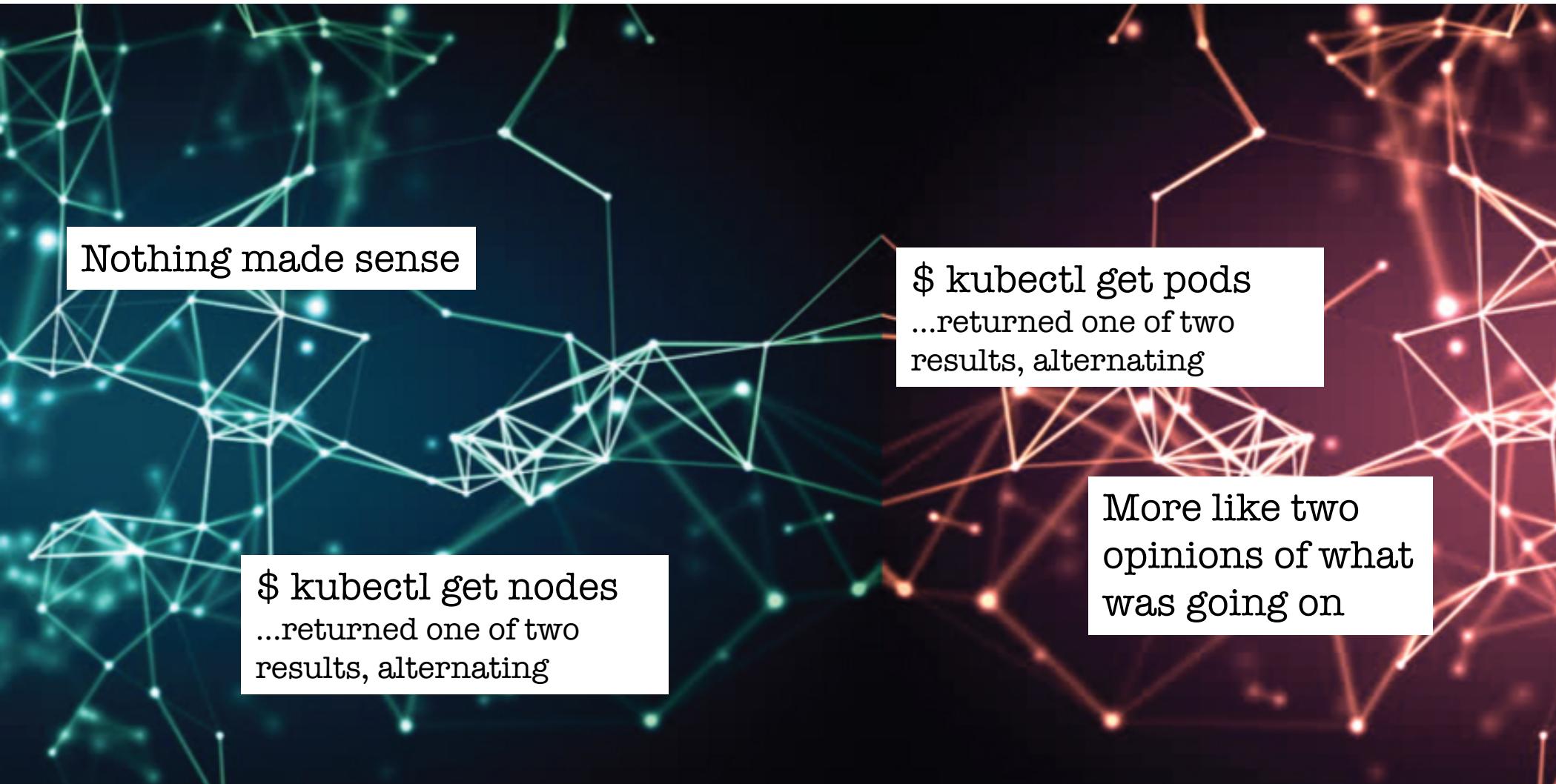
The [Cluster Autoscaler] Robots

- Along the way we learned some surprising things
 - Implicit session affinity of Kubernetes apiserver service (`kubernetes.default.svc.cluster.local`)
 - <https://github.com/kubernetes/kubernetes/pull/23129>
 - Disrespect of apiserver readiness when using HA config with `--apiserver-count` flag
 - <https://github.com/coreos/coreos-kubernetes/pull/730>



Lessons Learned

Split Personality etcd Cluster



Split Personality etcd Cluster

- This is not a conversation one looks forward to:



4:17 PM

@brett_wagner We're seeing a bit of odd behavior with some of our pods where the a new pod seems to start up and vanish, and start up and vanish... I've got `watch kubectl get pods` going and it's weird.

Split Personality etcd Cluster

- Especially not when you go look, and see:

Every 2.0s: kubectl get pods					Mk	.nordstrom.net: Thu Jun 29	2017
NAME	READY	STATUS	RESTARTS	AGE			
kubernetes-dashboard-2039414953-ct529	1/1	Running	0	6d			
kubernetes-dashboard-2039414953-h4xl2	0/1	Pending	0	5m			
oneoff-4068667279-172dt	1/1	Running	0	21d			
oneoff-4068667279-c1bkn	0/1	Pending	0	5m			
reviews-3048776615-7dzmp	2/2	Running	0	3d			

Every 2.0s: kubectl get pods					M	.nordstrom.net: Thu Jun 29	2017
NAME	READY	STATUS	RESTARTS	AGE			
kubernetes-dashboard-2039414953-h4xl2	0/1	Pending	0	12m			
oneoff-4068667279-c1bkn	0/1	Pending	0	13m			
reviews-3048776615-7dzmp	2/2	Running	0	3d			
reviews-3048776615-93z6l	2/2	Running	0	3d			
reviews-3048776615-nnz37	2/2	Running	0	3d			

Split Personality etcd Cluster

- The control loops started misbehaving
 - Thousands of pods
 - Many pending
 - Many terminating
 - Service endpoints thrashing
 - Ingress controller starting to do bad things



Split Personality etcd Cluster

- Bad news
 - Full cluster outage on primary production cluster
 - Not simply out of service, but violently wrong
 - Time-to-resolution was long: four hours
 - Spent time troubleshooting/diagnosing
 - Then replacing the cluster
 - Provisioning the replacement cluster was only the first step
 - Volumes were challenging
 - needed to release from old cluster, rebind on new
 - Cloud load balancers also challenging
 - ephemeral LB names were referenced in manually-managed DNS
 - migrating LBs across clusters not supported



Lessons Learned

Split Personality etcd Cluster

- Good news
 - Happened during working hours
 - Full team presence
 - Able to analyze and resolve root cause
 - Led to significant improvements in understanding, code, and procedures



Lessons Learned

Split Personality etcd Cluster

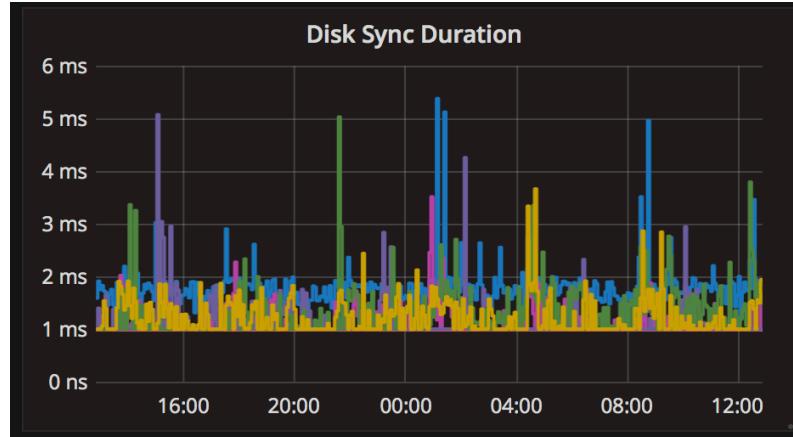
- But wait, etcd is a **consistent** k/v store, right?
- Yes, but...
 - It will happily return stale data (when configured to do so)
 - Stale data can happen multiple ways
 - This is documented (we had even RTFM!)
- The Kubernetes community was not in agreement that quorum reads were needed (or even desirable)
 - Not mentioned in HA docs (until Oct 2017)
 - Concerns about performance (etcd3 reduces impact)
 - And soon (in Kubernetes 1.9) quorum reads will be the default behavior



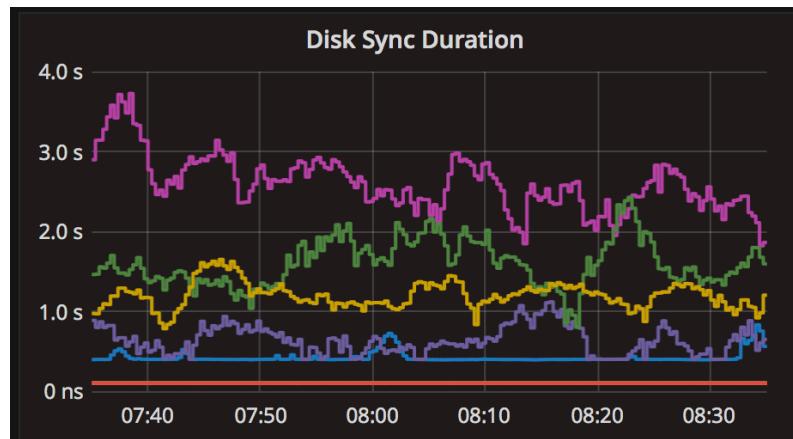
Lessons Learned

Split Personality etcd Cluster

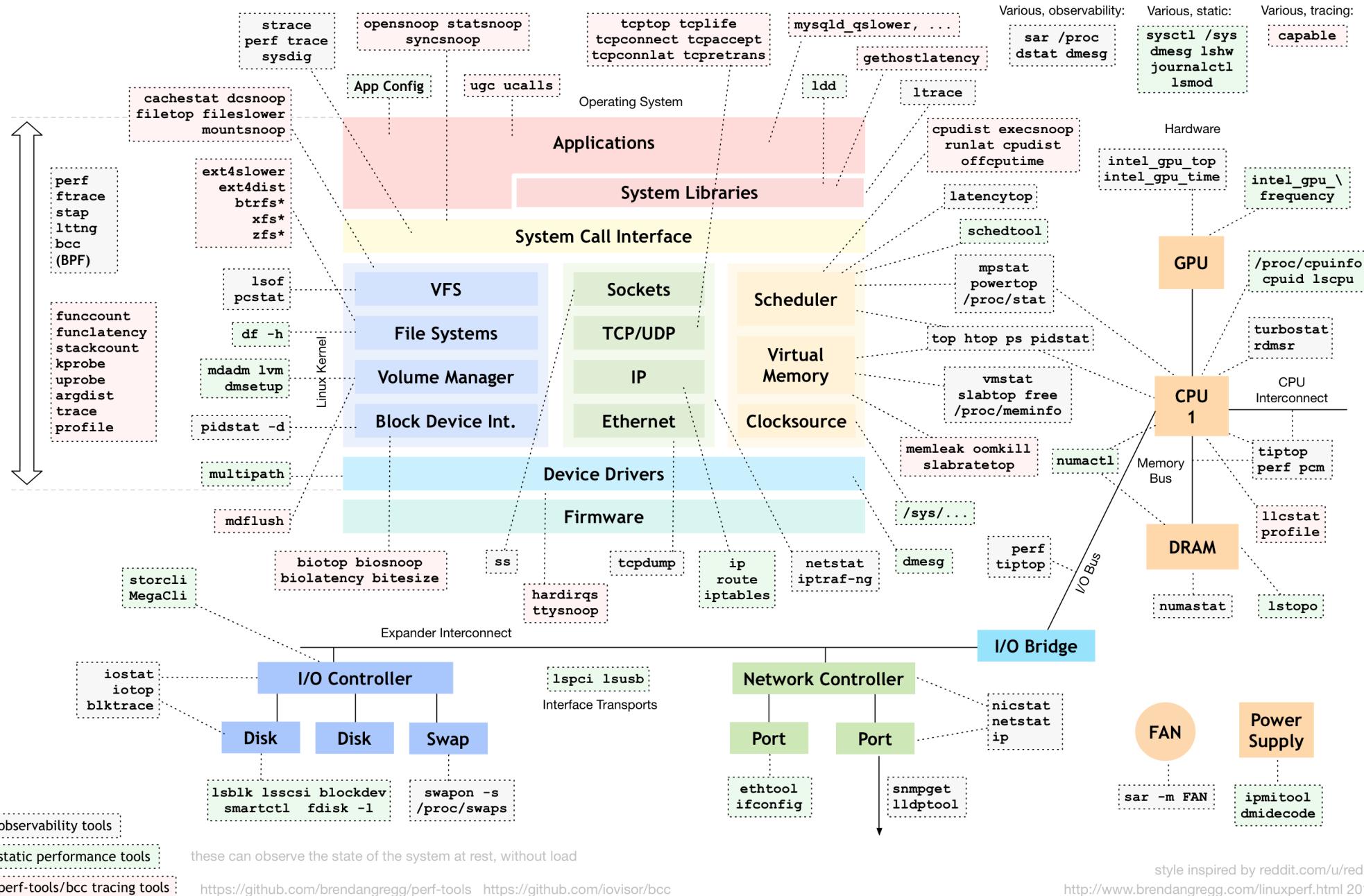
- Write latency is very important
 - This is healthy:



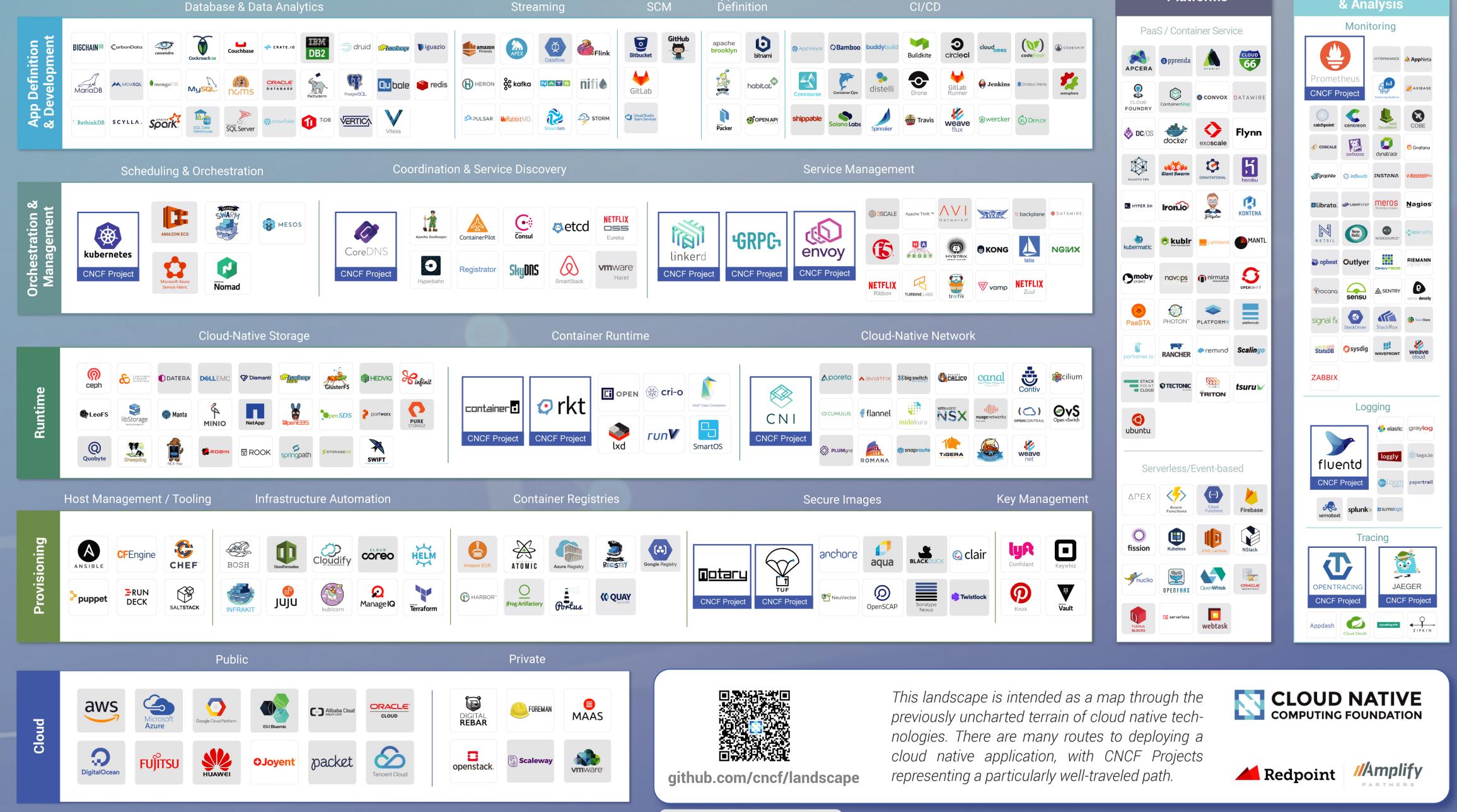
- This is not:



Lessons Learned



Cloud Native Landscape



Notes and references

- Problems with --apiserver-count flag
 - <https://github.com/kubernetes/kubernetes/issues/22609>
 - Fixed by “lease endpoint reconciler” in 1.9:
<https://github.com/kubernetes/kubernetes/pull/51698>
- Kyle Kingsbury’s Jepsen test of etcd, which led to quorum reads
 - <https://aphyr.com/posts/316-jepsen-etcd-and-consul>
- Discussion about Kubernetes apiserver using quorum reads
 - <https://github.com/kubernetes/kubernetes/issues/19902>
- Brendan Gregg on Linux performance
 - <http://www.brendangregg.com/linuxperf.html>



KubeCon

North America 2017
