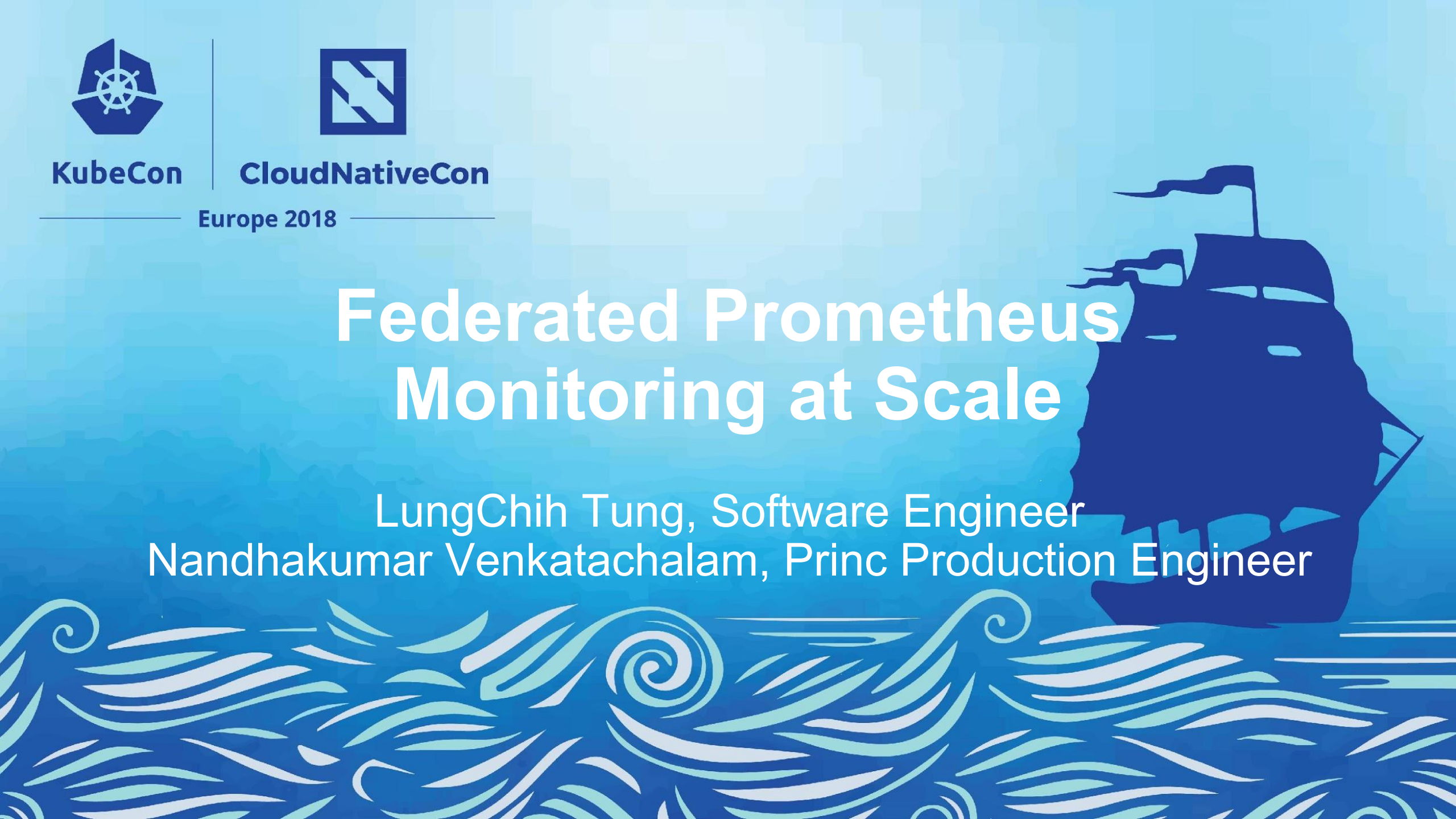
KubeCon | CloudNativeCon

Europe 2018

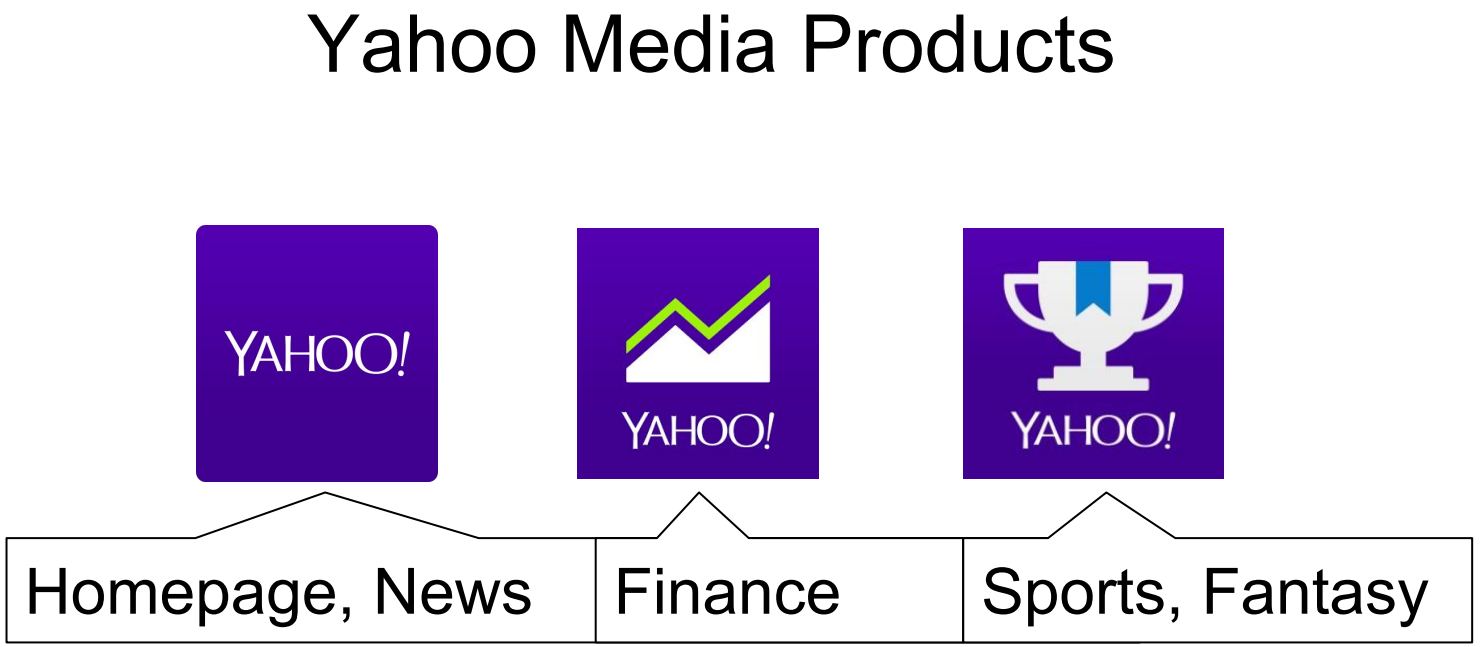
Federated Prometheus Monitoring at Scale

LungChih Tung, Software Engineer
Nandhakumar Venkatachalam, Princ Production Engineer



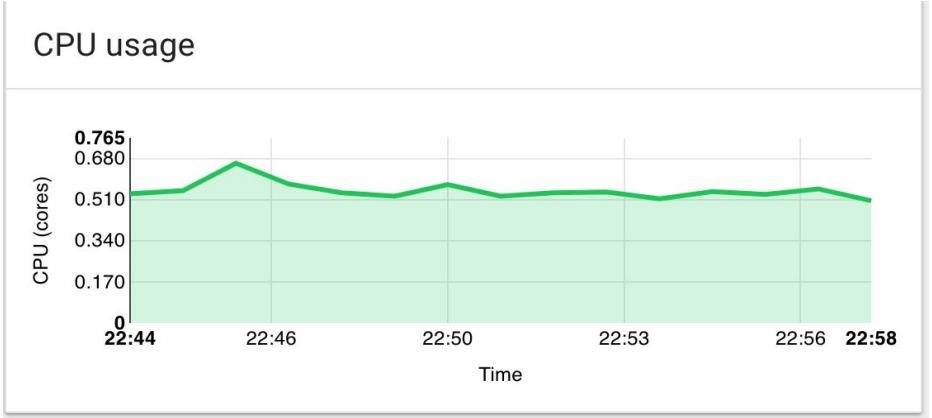
Team

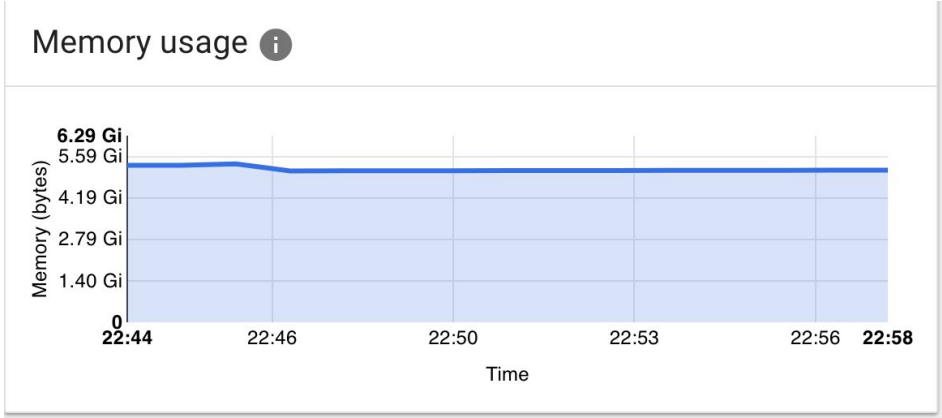
- Core Infrastructure
  - Infrastructure team powering all Yahoo Media Products



Journey to Kubernetes

STAGE colo1	ETCD	Master	Nodes			
CANARY colo1	ETCD	Master	Nodes			
PROD colo1	ETCD	Master	Nodes			

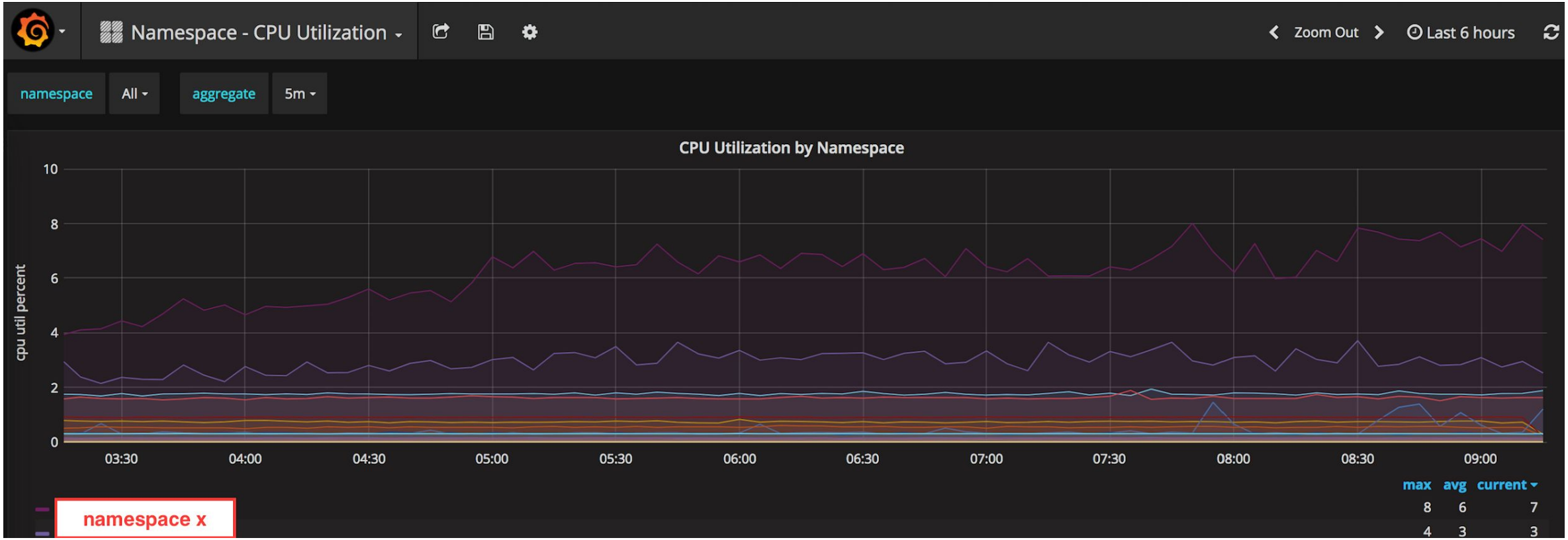




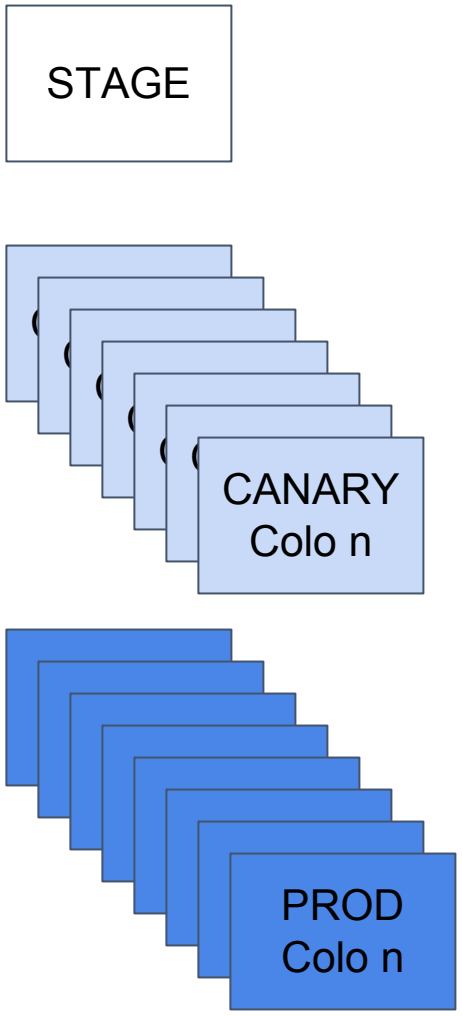
Monitoring Solutions

- Inhouse monitoring to get system's health
- Missing insight into kubernetes cluster metrics
- Heapster InFluxDB sink with grafana
- Prometheus 1.x
  - Remote write data into inhouse monitoring system

Heapster with InfluxDB



Growing Kubernetes Clusters



- Spanning across data centers
- Demand for higher visibility
- Growing monitoring requirement

Our Requirement

Cluster Health:

- ETCD
- Controller Manager
- Scheduler
- Kubernetes API server
- Kubelet
- Kubelet CAdvisor
- Kube DNS
- Any Add-ons...

Application Health:

- Namespace
- Deployment
- Pod
- Container

Prometheus 2.0 to the Rescue

- Huge performance improvement specifically in storage
- Simple syntax for aggregation and alerting rules
- Good documentation
- Our focus on pulling every metrics
- Aggregation rules

Metrics Collection

- Prometheus example yaml
- Endpoint discovery is simple.
  - annotation prometheus.io/scrape=true
- Simple File based discovery for apiservers

Metrics Collection - API Server


```
- job_name: kubernetes-apiservers
  scrape_interval: 30s
  scrape_timeout: 5s
  metrics_path: /metrics
  scheme: https
  file_sd_configs:
  - files:
    - /etc/prometheus/targets/apiserver-canary1bf1.json
    refresh_interval: 5m
  bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
  tls_config:
    ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
    insecure_skip_verify: false
  relabel_configs:
  - separator: ;
    regex: (.*)
    target_label: colo
    replacement: bf1
    action: replace
```

Metrics Proxy

- ETCD ports are accessible only by master machines
- Scheduler and controller ports bind to 127.0.0.1

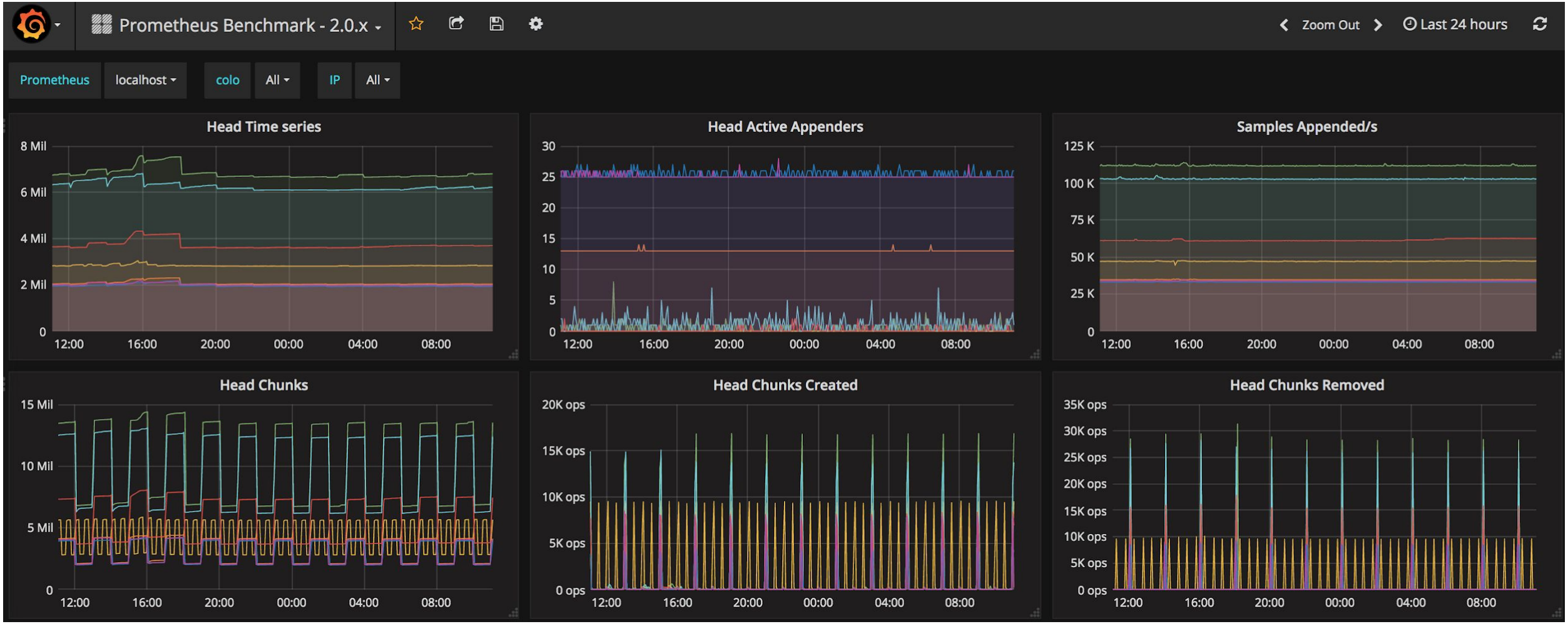
Metrics Proxy - ETCD


```
- job_name: kubernetes-etcd
  params:
    port:
    - "2379"
  scrape_interval: 30s
  scrape_timeout: 5s
  metrics_path: /metrics
  scheme: https
  file_sd_configs:
  - files:
    - /etc/prometheus/targets/etcd-canary1bf1.json
    refresh_interval: 5m
  bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
  tls_config:
    ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
    insecure_skip_verify: false
  relabel_configs:
  - separator: ;
    regex: (.*)
    target_label: colo
    replacement: bf1
    action: replace
```

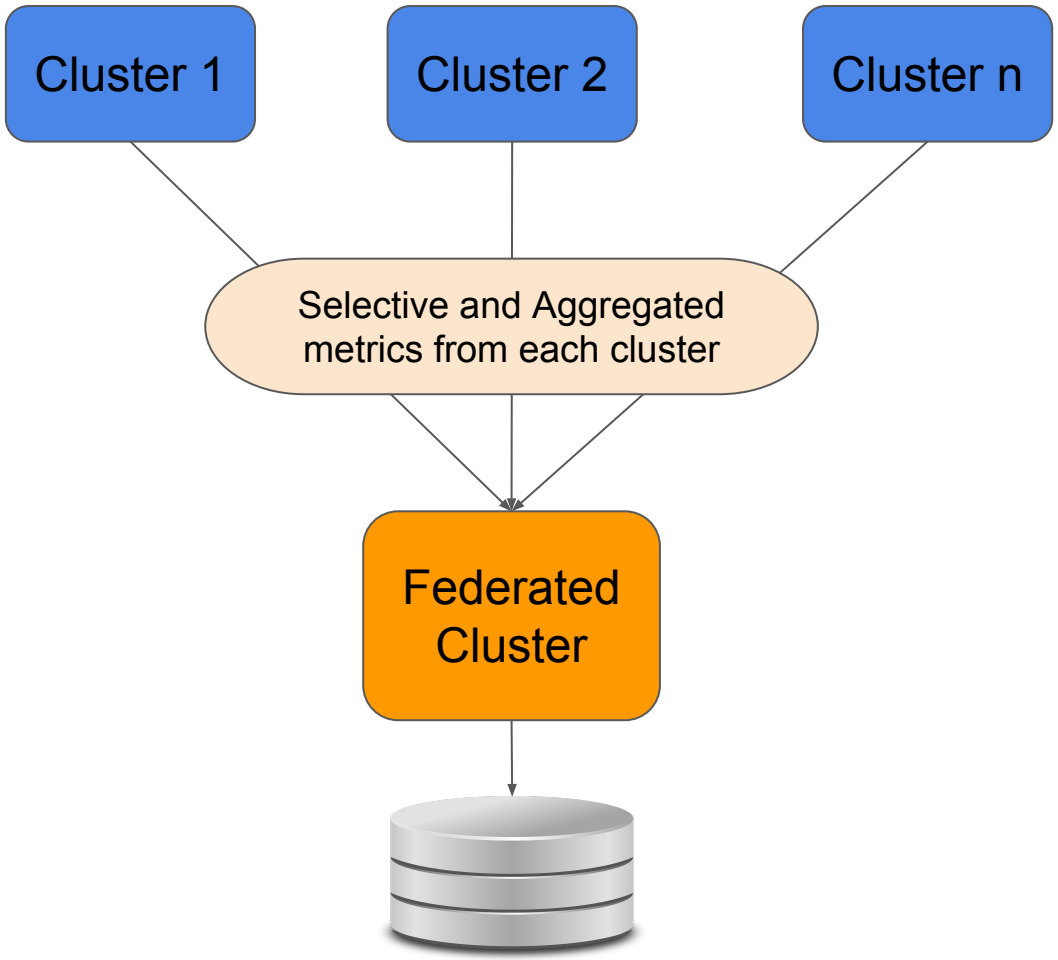
Metrics Proxy - Controller

```
- job_name: kubernetes-controller
  params:
    port:
    - "10252"
  scrape_interval: 30s
  scrape_timeout: 5s
  metrics_path: /metrics
  scheme: https
  file_sd_configs:
  - files:
    - /etc/prometheus/targets/controller-canary1bf1.json
    refresh_interval: 5m
  bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
  tls_config:
    ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
    insecure_skip_verify: false
  relabel_configs:
  - separator: ;
    regex: (.*)
    target_label: colo
    replacement: bf1
    action: replace
```

Volume of the metrics



Federation



- Aggregated time series data
- Longer retention period
- Permanent storage
- Unified display of data

Federation Configuration

- Federate all control plane components
- Selective aggregated metrics

```
- job_name: 'federate'

  honor_labels: true
  metrics_path: '/federate'

  params:
    'match[]':
      - '{job="prometheus"}'
      - '{app="kube-state-metrics"}'
      - '{job="kubernetes-etcd"}'
      - '{job="kube-dns"}'
      - '{job="kubernetes-scheduler"}'
      - '{job="kubernetes-controller"}'
      - '{job="kubernetes-nodes"}'
      - '{job="kubernetes-apiservers"}'
      - '{__name__=~"cluster.*"}'
      - '{__name__=~"colo.*"}'
      - '{kubernetes_name="heapster"}'

  static_configs:
    - targets:
```

Aggregation Rules

- Ton of rules from prometheus operator team
- Built CPU and Memory utilization by
  - Colo(cluster) level
  - Namespace/ Deployment/ Pod and Container level

Aggregation - Colo level


```
# colo level cpu utilization
- record: colo:cpu_percentage:rate
   expr: 100 *
sum(label_replace(irate(container_cpu_usage_seconds_total{container_name!="",
container_name!="POD"}[5m]), "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo)
/
sum(label_replace(container_spec_cpu_shares{container_name!="",
container_name!="POD"}, "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo) * 1000
```

CPU usage of all containers per colo

CPU shares allocated to all containers per colo

Aggregation - Namespace level


```
# namespace level cpu utilization
- record: colo_namespace:cpu_percentage:rate
   expr: 100 *
sum(label_replace(irate(container_cpu_usage_seconds_total{container_name!="",
container_name!="POD"}[5m]), "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo, namespace)
/
sum(label_replace(container_spec_cpu_shares{container_name!="",
container_name!="POD"}, "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY (colo,
namespace) * 1000
```

CPU usage of all containers per namespace

CPU shares allocated to all containers per namespace

Aggregation - Controller level


```
# controller level cpu utilization
- record: colo_namespace_controller:cpu_percentage:rate
   expr: 100 *
sum(label_replace(irate(container_cpu_usage_seconds_total{container_name!="",
container_name!="POD"}[5m]), "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo, namespace, controller)
/
sum(label_replace(container_spec_cpu_shares{container_name!="",
container_name!="POD"}, "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY (colo,
namespace, controller) * 1000
```

CPU usage of all containers per controller

CPU shares allocated to all containers per controller

Aggregation - Pod level


```
# pod level cpu utilization
- record: colo_namespace_controller_pod:cpu_percentage:rate
expr: 100 *
sum(label_replace(irate(container_cpu_usage_seconds_total{container_name!="",
container_name!="POD"}[5m]), "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo, namespace, controller, pod_name)
/
sum(label_replace(container_spec_cpu_shares{container_name!="",
container_name!="POD"}, "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo, namespace, controller, pod_name) * 1000
```

CPU usage of all containers per pod

CPU shares allocated to all containers per pod

Aggregation - Container level


```
# container level cpu utilization
- record: colo_namespace_controller_pod_container:cpu_percentage:rate
   expr: 100 *
sum(label_replace(irate(container_cpu_usage_seconds_total{container_name!="",
container_name!="POD"}[5m]), "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo, namespace, controller, pod_name, container_name)
/
sum(label_replace(container_spec_cpu_shares{container_name!="",
container_name!="POD"}, "controller", "$1", "pod_name", "^(.*)-[a-z0-9]+")) BY
(colo, namespace, controller, pod_name, container_name) * 1000
```

CPU usage per container

CPU shares allocated to container

Alert Manager

- Alert manager defines how to handle alerts (email, slack notification, etc.)
  - Grouping alert
  - Silences
  - Inhibition

Alert Rules


```
- alert: K8SNodeNotReady
  expr: kube_node_status_condition{condition="Ready",status="true"} == 0
  for: 1h
  labels:
    severity: warning
    colo: bf1
    environment: production
  annotations:
    description: The Kubelet on {{ $labels.node }} has not checked in with the API,
      or has set itself to NotReady, for more than an hour
    summary: Node status is NotReady
```

Alert Rules


```
- alert: PodCPUPercentage
  expr: colo_namespace_controller_pod:cpu_percentage:rate{namespace=~"kube-.*"} > 75
  for: 10m
  labels:
    severity: critical
    colo: bf1
    environment: production
  annotations:
    description: 'Pod cpu usage is above 75 for {{ $value }}.
      Please find out the cause of the spike and if required increase CPU allocation'
    summary: Pod cpu usage is above threshold
```

Alerting on Prometheus

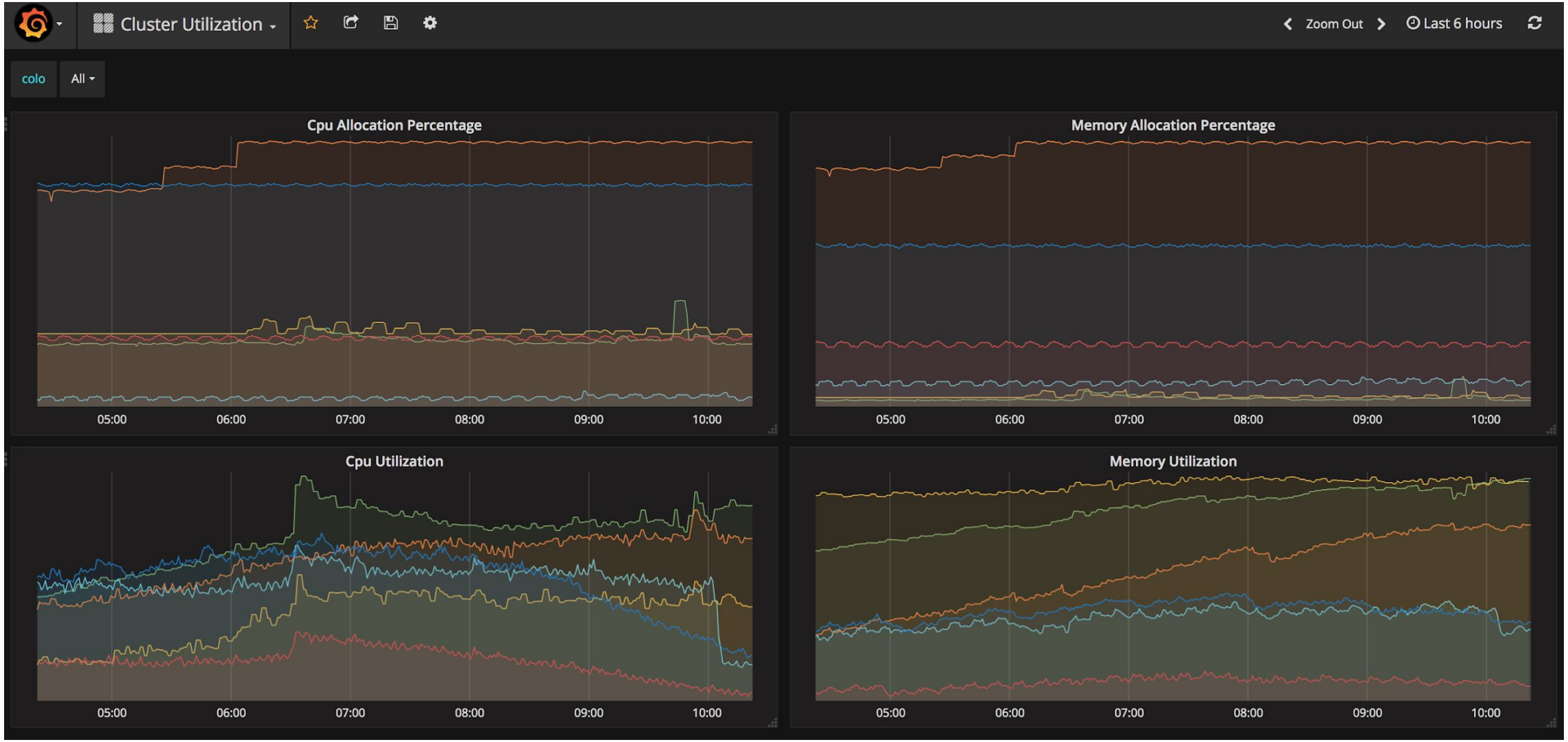
- Federated prometheus monitors individual prometheus
- A cron job monitors federated prometheus

PrometheusDown (0 active)

```
alert: PrometheusDown
expr: up{job="federate"}
  == 0
for: 15m
labels:
  colo: federation
  environment: production
  severity: critical
annotations:
  description: Prometheus {{ $labels.instance }} are down. Please check the pod status
    and error log by running kubectl. Prometheus uses a lot of memory, so also check
    the memory usage of prometheus in kube dashboard. If prometheus is not schedulale
    due to lack of memory, try to fix a down node or drain a node to make room for
    it.
  summary: Prometheus are down
```

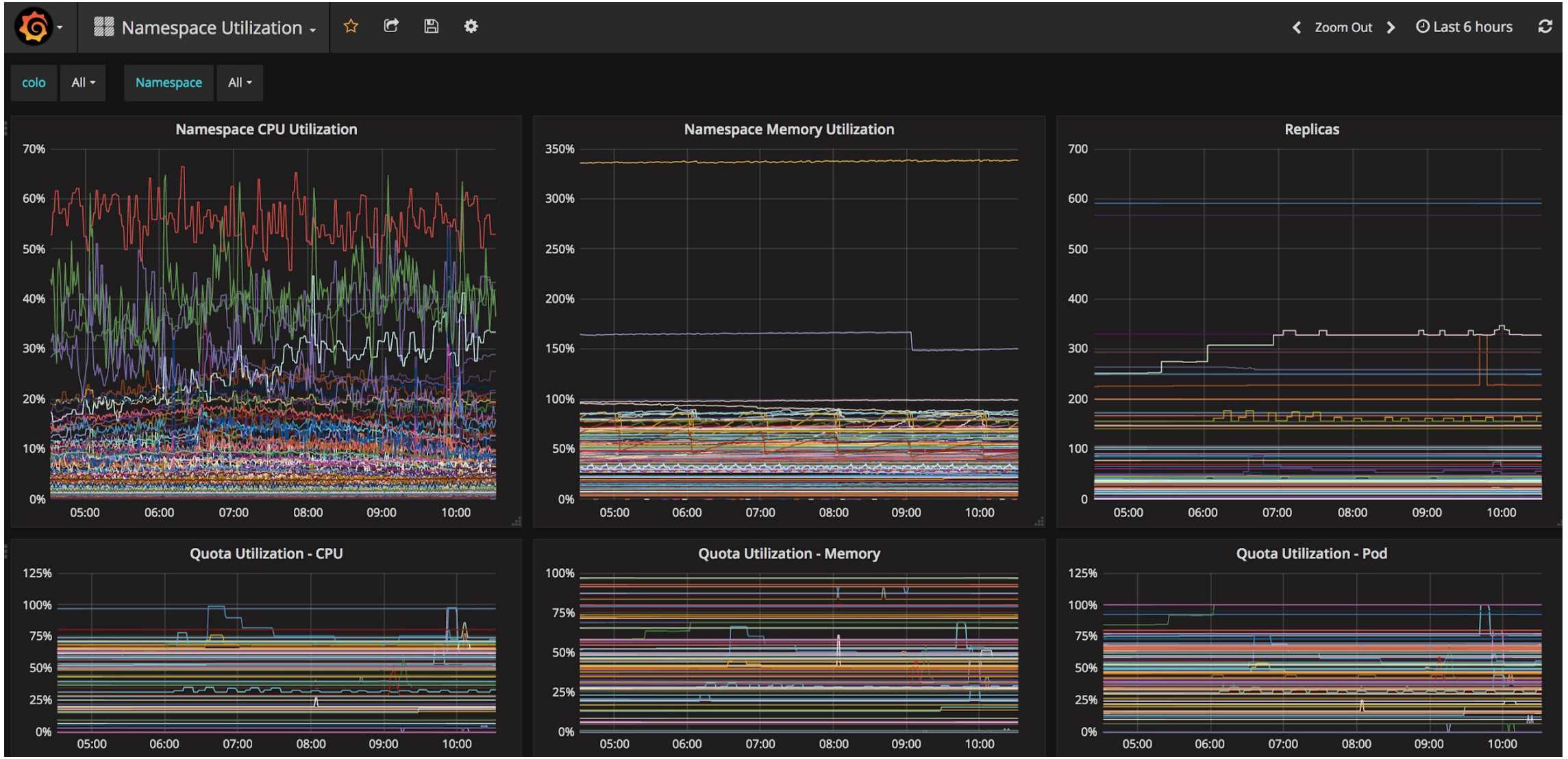
KubeCon
CloudNativeCon
Europe 2018
Dashboards

Cluster View

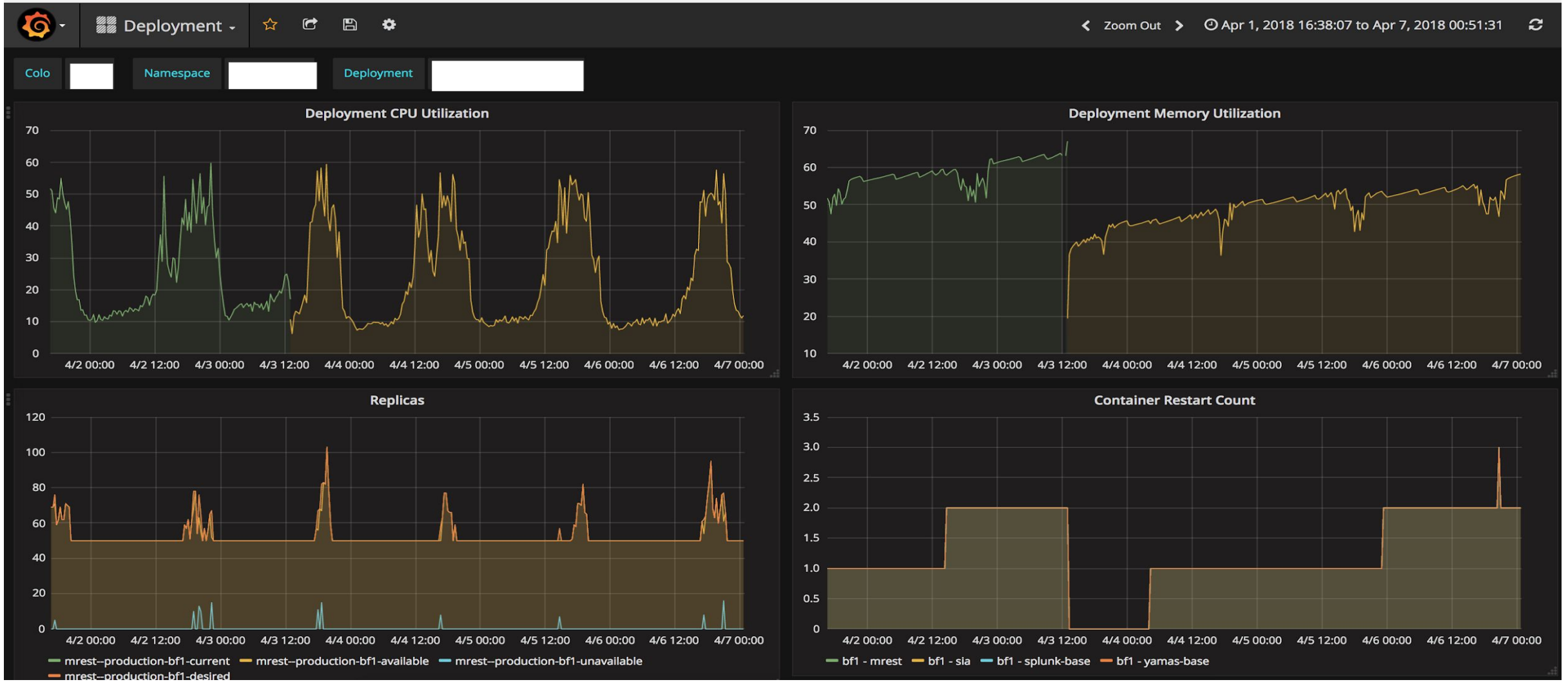


Namespace

KubeCon | CloudNativeCon
Europe 2018



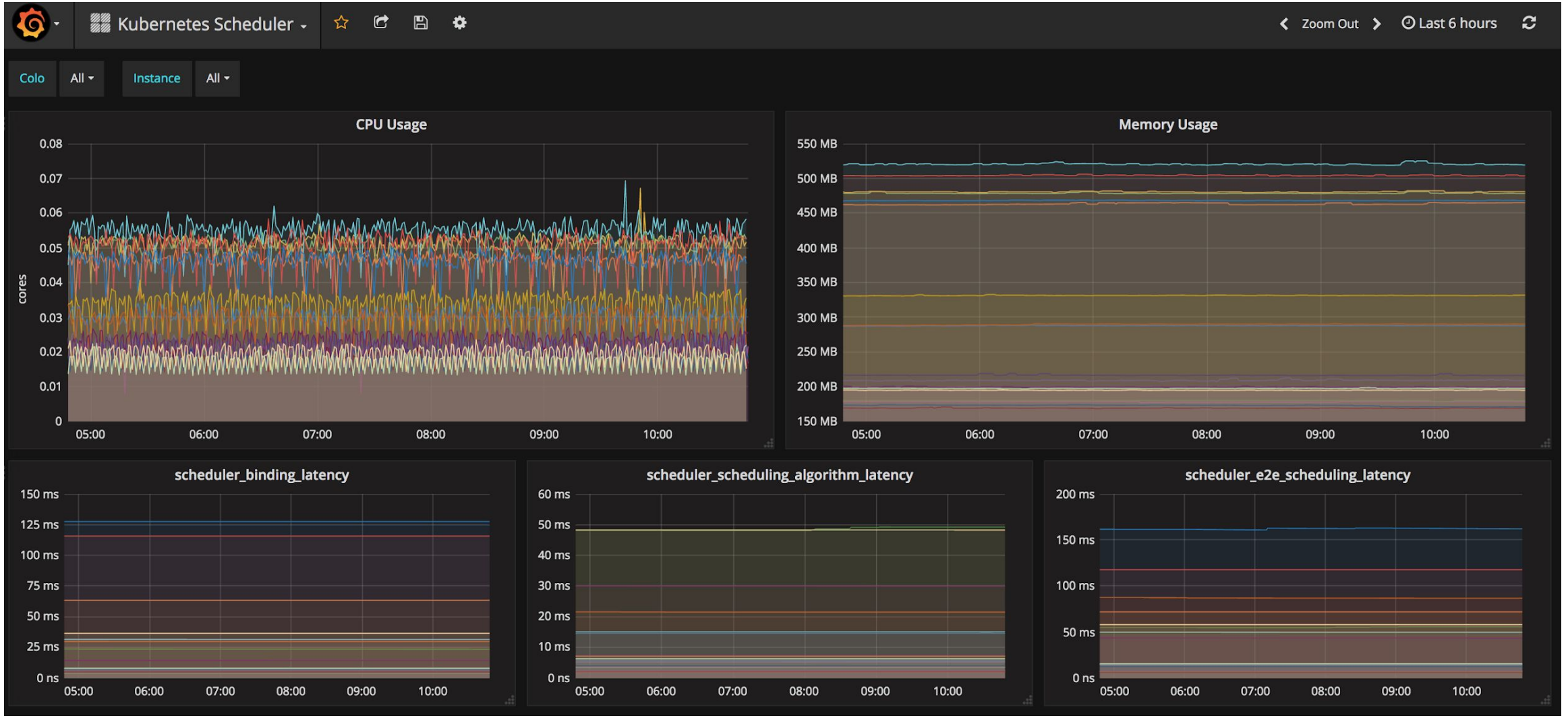
Deployment


Deployment
Zoom Out
Apr 1, 2018 16:38:07 to Apr 7, 2018 00:51:31
Colo
Namespace
Deployment
Deployment CPU Utilization
70
60
50
40
30
20
10
0
4/2 00:00
4/2 12:00
4/3 00:00
4/3 12:00
4/4 00:00
4/4 12:00
4/5 00:00
4/5 12:00
4/6 00:00
4/6 12:00
4/7 00:00
Deployment Memory Utilization
70
60
50
40
30
20
10
4/2 00:00
4/2 12:00
4/3 00:00
4/3 12:00
4/4 00:00
4/4 12:00
4/5 00:00
4/5 12:00
4/6 00:00
4/6 12:00
4/7 00:00
Replicas
120
100
80
60
40
20
0
4/2 00:00
4/2 12:00
4/3 00:00
4/3 12:00
4/4 00:00
4/4 12:00
4/5 00:00
4/5 12:00
4/6 00:00
4/6 12:00
4/7 00:00
mrest--production-bf1-current
mrest--production-bf1-available
mrest--production-bf1-unavailable
mrest--production-bf1-desired
Container Restart Count
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0
4/2 00:00
4/2 12:00
4/3 00:00
4/3 12:00
4/4 00:00
4/4 12:00
4/5 00:00
4/5 12:00
4/6 00:00
4/6 12:00
4/7 00:00
bf1 - mrest
bf1 - sia
bf1 - splunk-base
bf1 - yamas-base

Controller

Kubernetes Controller
Zoom Out
Last 5 minutes
Colo
All
Instance
All
CPU Usage
cores
0.30
0.25
0.20
0.15
0.10
0.05
0
10:32:00 10:32:30 10:33:00 10:33:30 10:34:00 10:34:30 10:35:00 10:35:30 10:36:00 10:36:30
Memory Usage
4.0 GB
3.0 GB
2.0 GB
1.0 GB
0 B
10:32:00 10:32:30 10:33:00 10:33:30 10:34:00 10:34:30 10:35:00 10:35:30 10:36:00 10:36:30
Horizontal pod autoscaler queue latency
30 Mil
25 Mil
20 Mil
15 Mil
10 Mil
5 Mil
0
10:32:00 10:32:30 10:33:00 10:33:30 10:34:00 10:34:30 10:35:00 10:35:30 10:36:00 10:36:30
Horizontal pod autoscaler work duration
800 K
600 K
400 K
200 K
0
10:32:00 10:32:30 10:33:00 10:33:30 10:34:00 10:34:30 10:35:00 10:35:30 10:36:00 10:36:30
Deployment queue latency
60
50
40
30
20
10
10:32:00 10:32:30 10:33:00 10:33:30 10:34:00 10:34:30 10:35:00 10:35:30 10:36:00 10:36:30
Deployment work duration
10.0 K
9.0 K
8.0 K
7.0 K
6.0 K
5.0 K
4.0 K
10:32:00 10:32:30 10:33:00 10:33:30 10:34:00 10:34:30 10:35:00 10:35:30 10:36:00 10:36:30

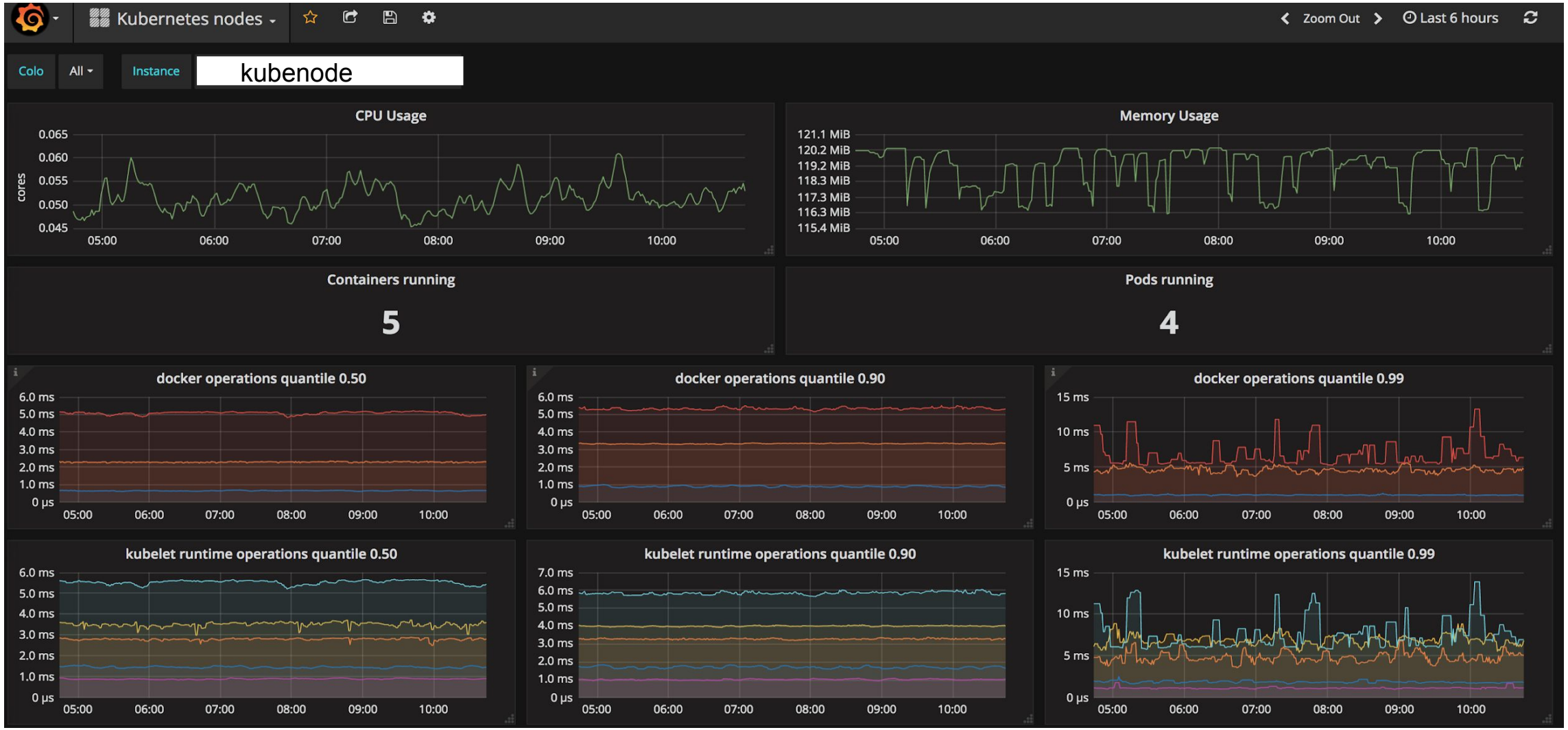
Scheduler


Kubernetes Scheduler
Zoom Out
Last 6 hours
Colo
All
Instance
All
CPU Usage
cores
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
05:00
06:00
07:00
08:00
09:00
10:00
Memory Usage
550 MB
500 MB
450 MB
400 MB
350 MB
300 MB
250 MB
200 MB
150 MB
05:00
06:00
07:00
08:00
09:00
10:00
scheduler_binding_latency
150 ms
125 ms
100 ms
75 ms
50 ms
25 ms
0 ns
05:00
06:00
07:00
08:00
09:00
10:00
scheduler_scheduling_algorithm_latency
60 ms
50 ms
40 ms
30 ms
20 ms
10 ms
0 ns
05:00
06:00
07:00
08:00
09:00
10:00
scheduler_e2e_scheduling_latency
200 ms
150 ms
100 ms
50 ms
0 ns
05:00
06:00
07:00
08:00
09:00
10:00

API server

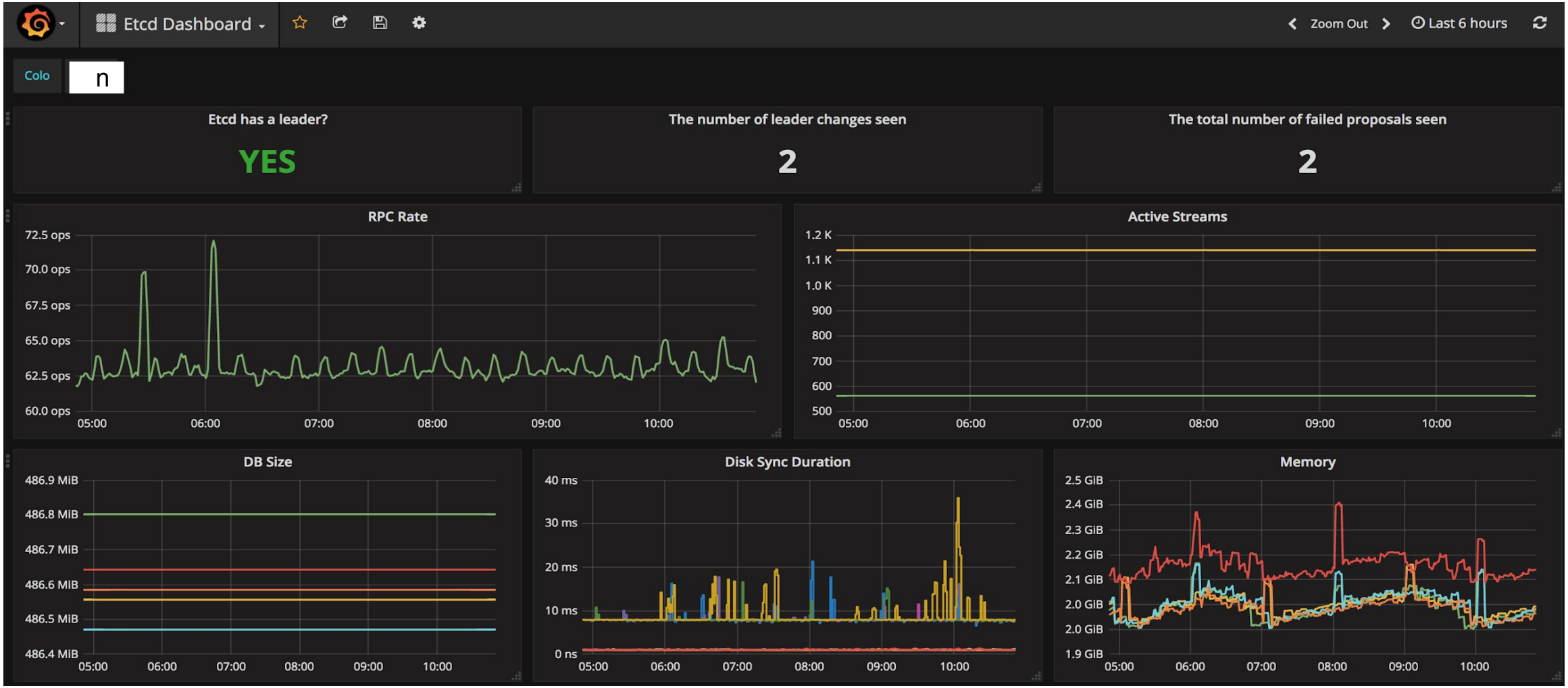
Kubernetes API
Zoom Out
Last 24 hours
Colo
All
Instance
All
client
All
resource
All
CPU Usage
1.50 s
1.25 s
1.00 s
750 ms
500 ms
250 ms
0 ns
12:00
16:00
20:00
00:00
04:00
08:00
Memory Usage
50 MB
40 MB
30 MB
20 MB
10 MB
0 B
12:00
16:00
20:00
00:00
04:00
08:00
Request Rate by Client
400
300
200
100
0
12:00
16:00
20:00
00:00
04:00
08:00
Latency Resource
80 Bil
60 Bil
40 Bil
20 Bil
0
12:00
16:00
20:00
00:00
04:00
08:00

Kubelet

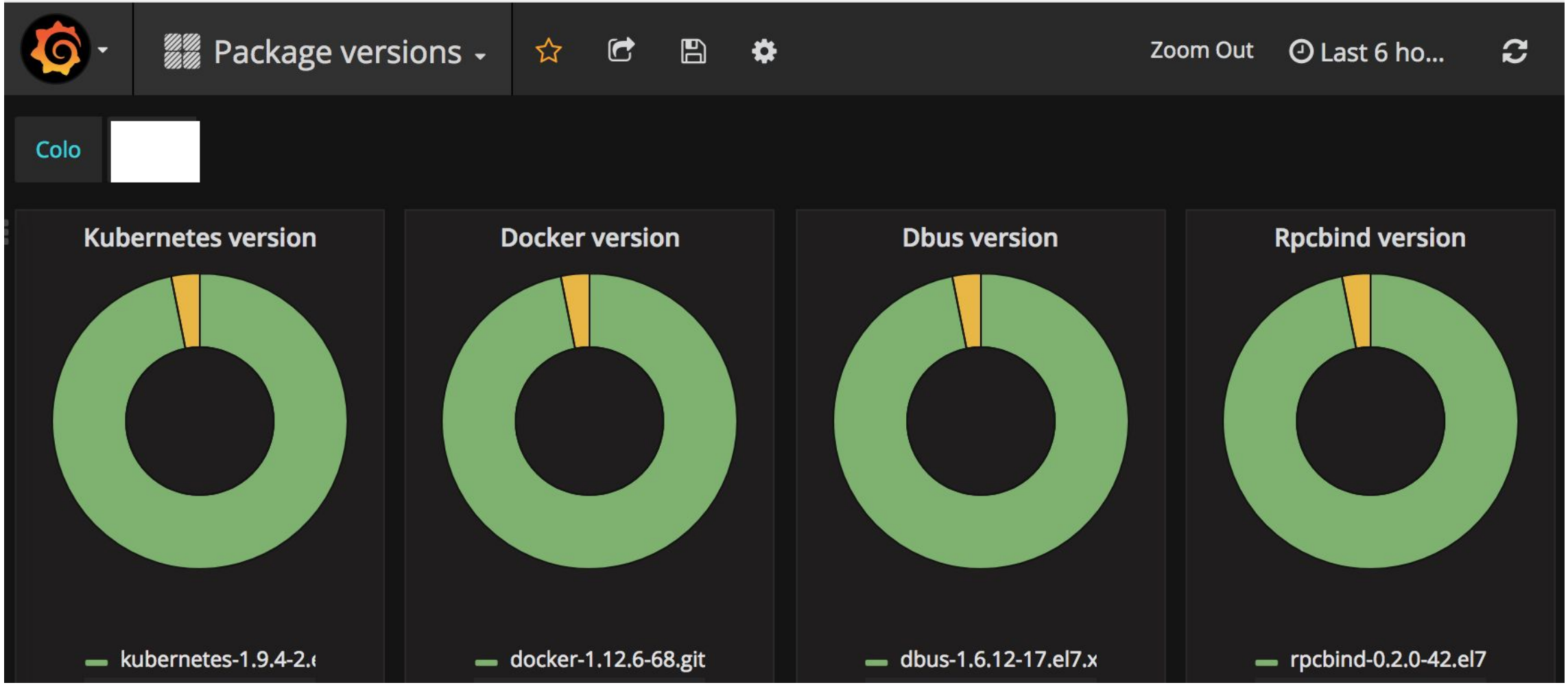

Kubernetes nodes
Zoom Out
Last 6 hours
Colo
All
Instance
kubenode
CPU Usage
cores
0.065
0.060
0.055
0.050
0.045
05:00
06:00
07:00
08:00
09:00
10:00
Memory Usage
121.1 MiB
120.2 MiB
119.2 MiB
118.3 MiB
117.3 MiB
116.3 MiB
115.4 MiB
Containers running
5
Pods running
4
docker operations quantile 0.50
6.0 ms
5.0 ms
4.0 ms
3.0 ms
2.0 ms
1.0 ms
0 µs
docker operations quantile 0.90
docker operations quantile 0.99
15 ms
10 ms
5 ms
kubelet runtime operations quantile 0.50
kubelet runtime operations quantile 0.90
7.0 ms
kubelet runtime operations quantile 0.99

ETCD

KubeCon | CloudNativeCon Europe 2018



Package version



Thank you

KubeCon

CloudNativeCon

Europe 2018

Q & A