Observability Day
NORTH AMERICA

PROMETHEUS

QUERIES THAT MEASURE CPU AND MEMORY

⚡ Lightning Talk: Vote on Your Favorite--Standardize This!--10 PromQL Queries That Measure CPU & Memory Usage

AGENDA

01

INTRO

Me and why I love hate PromQL

02

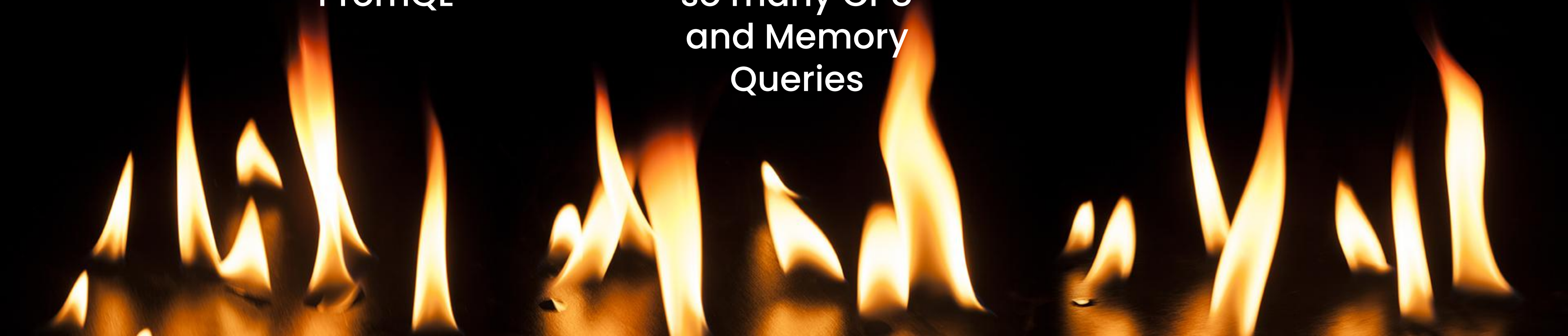
WHAT WHY WHERE WHEN

The what, why, when, where of so many CPU and Memory Queries

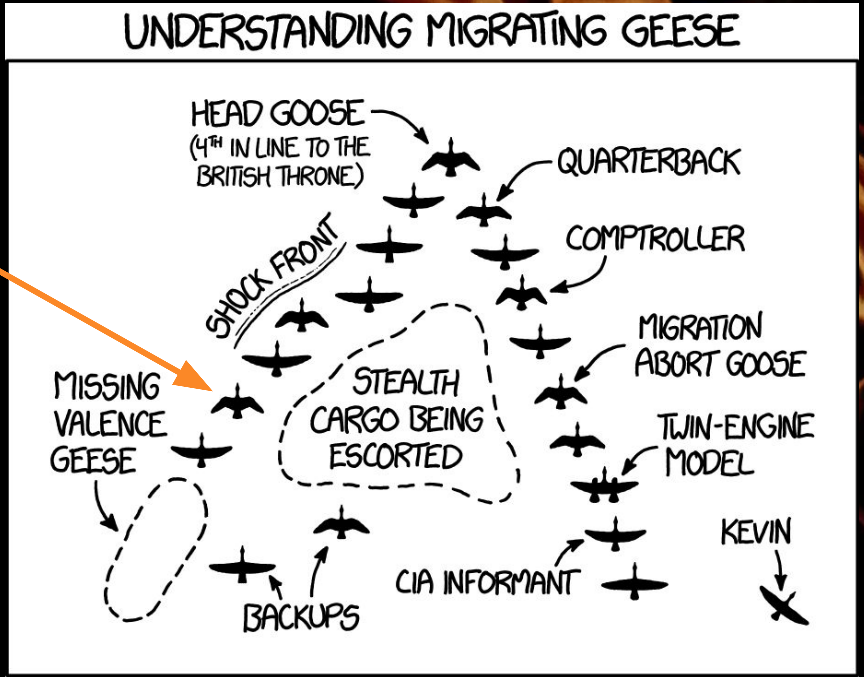
03

VOTE NOW!

PICK your favorite!



ME

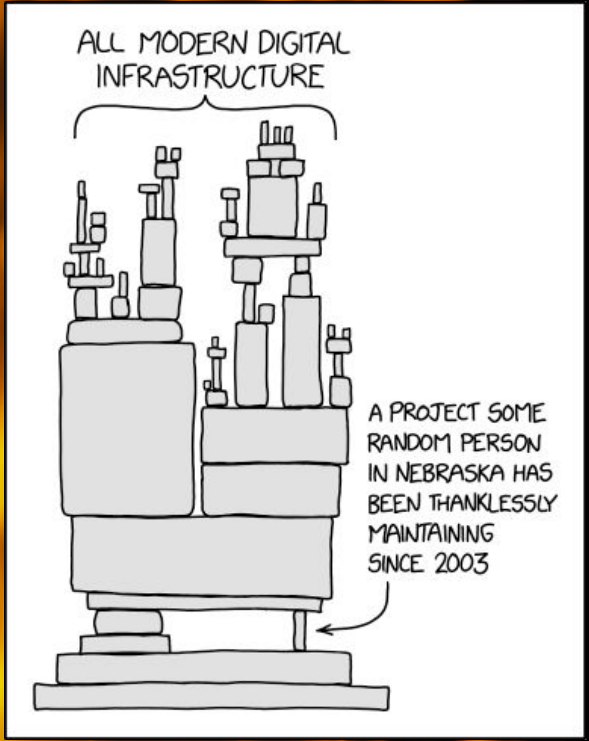


Solutions Architect Migration Work to PROMQL

- *SignalFX to Prom*
- *Wavefront to Prom*
- *DogstatsD to Prom*
- *NRQL to Prom*

So many Queries

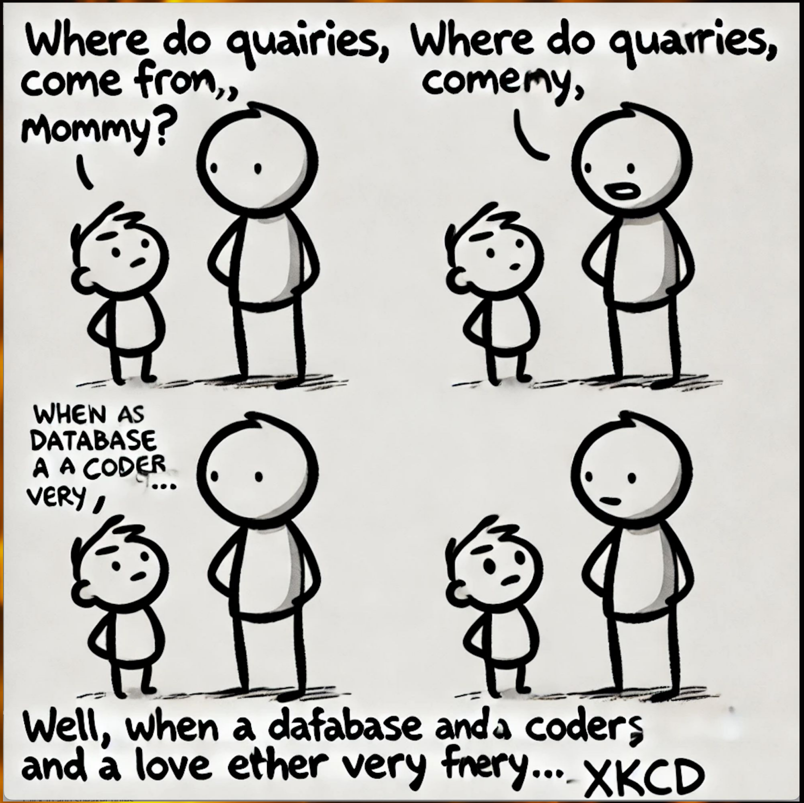
chronosphere


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003

THE WHY

MIGRATIONS

Change is hard


Where do quairies, come fron,, Mommy?
Where do quarries, comemy,
WHEN AS DATABASE A A CODER VERY, ...
Well, when a dafabase and a coders and a love ether very fnery... XKCD

THE WHERE

AI GENERATED EXPLANATION

"When a database and a coder love each other very much…

THREE MAIN SOURCES OF QUERIES

CUSTOMER TRANSLATIONS

```
(sum by (kubernetes_cluster) (
        rate(
container_cpu_usage_seconds_total{contai
ner="proxy",kubernetes_cluster=~"prod-.*
|dev|staging"}[30s]
        ) *100))
/
  (sum by (kubernetes_cluster) (
kube_pod_container_resource_requests{con
tainer="proxy",kubernetes_cluster=~"prod
-.*|dev|staging",resource="cpu"}
    ))
```

INTERNAL DASHBOARDS AND MONITORS

```
avg by (container, environment,
            az) (

rate(container_cpu_usage_second
s_total{container=~".*",instanc
     e=~".*",pod=~".*"}[5m])
             )
```

THE INTERNET

```
rate(container_cpu_usage_sec
onds_total{container!="",nam
espace="my-namespace",pod=~"
my-pod.*"}[15m])
```

THE WHAT

QUERIES

CUSTOMER TRANSLATION QUERIES

SIGNALFX/WAVEFRONT

```
1.   (sum by (kubernetes_cluster)
     (rate(container_cpu_usage_seconds_total{container="proxy",kubernetes_cluster=~"prod-.*|dev|staging"}[30s])
     * 100))/ (sum by (kubernetes_cluster)
     (kube_pod_container_resource_requests{kubernetes_cluster=~"prod-.*|dev|staging",container="proxy",resource
     ="cpu"}))   % Percentage

2.   max_over_time(kube_node_status_allocatable{resource="cpu",kubernetes_cluster=~".*prod.*"}[1h])  # of Cores

3.   count(sum by (pod)
     (avg_over_time(container_cpu_usage_seconds_total{kubernetes_cluster=~".*prod.*"}[$interval])))  Pod Count

4.   sum(rate(container_cpu_usage_seconds_total{namespace=~"$namespace",instance=~"$node",kubernetes_cluster=~"
     $kubernetes_cluster",container=~"$container"}[$interval]))Core Usage/per Second

5.   avg(rate(container_cpu_usage_seconds_total{namespace=~"$namespace",instance=~"$node",name!="",kubernetes_c
     luster=~"$kubernetes_cluster", pod=~"$pod"}[$interval]) / on(namespace, pod, id, kubernetes_cluster)
     ((container_spec_cpu_quota{namespace=~"$namespace",instance=~"$node",name!="",kubernetes_cluster=~"$kubern
     etes_cluster",pod=~"$pod"} > 0) / 100000)) by (namespace, pod, kubernetes_cluster)  % Percentage
```

PRETTY PROM

```
 (
    sum by (kubernetes_cluster) (
        rate(
          container_cpu_usage_seconds_total{container="proxy",kubernetes_cluster=~"prod-.*|dev|staging"}[30s]
        )
      *
        100
    )
  )
/
  (
    sum by (kubernetes_cluster) (
      kube_pod_container_resource_requests{container="proxy",kubernetes_cluster=~"prod-.*|dev|staging",resource="cpu"}
    )
  )
```

```
sum(
  rate(

container_cpu_usage_seconds_total{container=~"$container",instance=~"$node",kubernetes_cluster=~"$kubernetes_cluster",namespace=~"$namespa
ce"}[15m]
  )
)
```

```
avg by (namespace, pod, kubernetes_cluster) (
    rate(
      container_cpu_usage_seconds_total{instance=~"$node",kubernetes_cluster=~"$kubernetes_cluster",name!="",namespace=~"$namespace",pod=~"$pod"}[15m]
    )
  / on (namespace, pod, id, kubernetes_cluster)
    (
        (
            container_spec_cpu_quota{instance=~"$node",kubernetes_cluster=~"$kubernetes_cluster",name!="",namespace=~"$namespace",pod=~"$pod"}
          >
            0
        )
      /
        100000))
```

INTERNAL CHRONOSPHERE QUERIES

QUERIES PER PANEL WITH MULTIPLE PANELS PER DASHBOARD

6. CPU USAGE (cores)

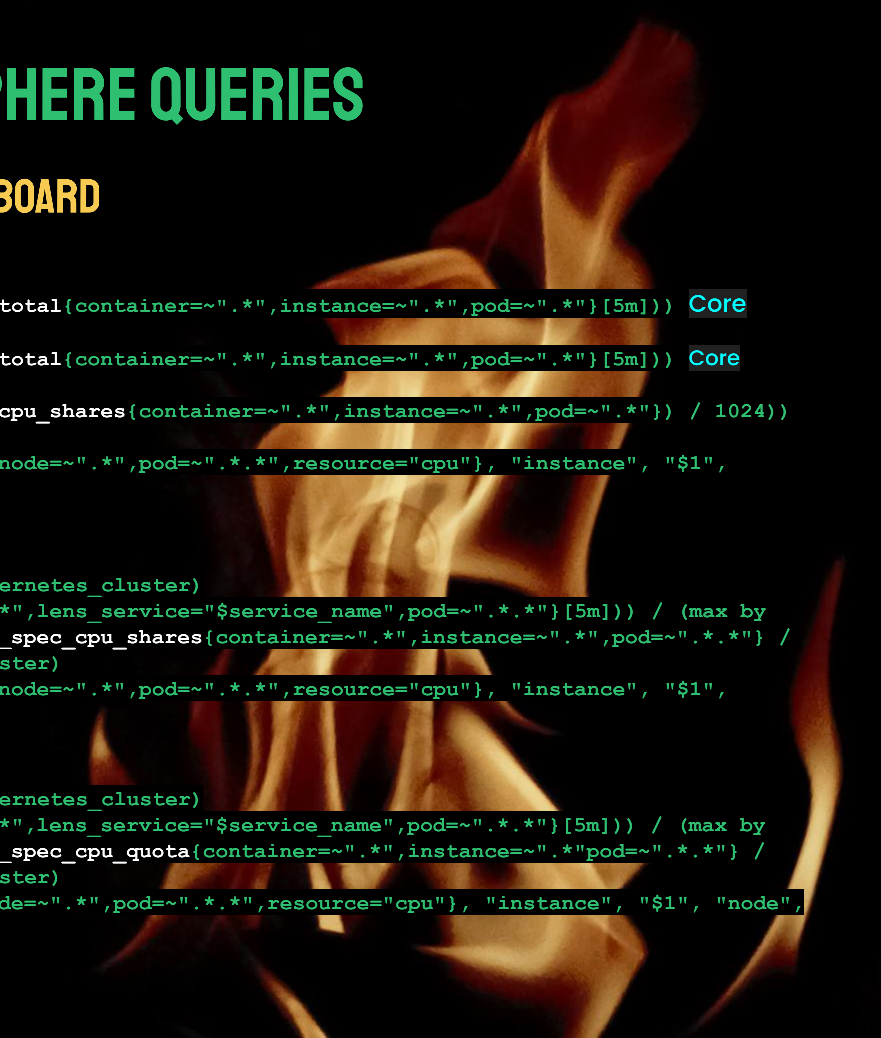
- `avg by (container,environment,az) (rate(container_cpu_usage_seconds_total{container=~".*",instance=~".*",pod=~".*"}[5m]))` Core usage/per second
- `max by (container,environment,az) (rate(container_cpu_usage_seconds_total{container=~".*",instance=~".*",pod=~".*"}[5m]))` Core usage/per second
- `(max by (container,environment,kubernetes_cluster) ((container_spec_cpu_shares{container=~".*",instance=~".*",pod=~".*"}) / 1024)) or (max by (container,environment,kubernetes_cluster) (label_replace(kube_pod_container_resource_requests{container=~".*",node=~".*",pod=~".*.*",resource="cpu"}, "instance", "$1", "node", "(.*)")))` # of Cores

7. CPU Usage as a Percentage of Requests

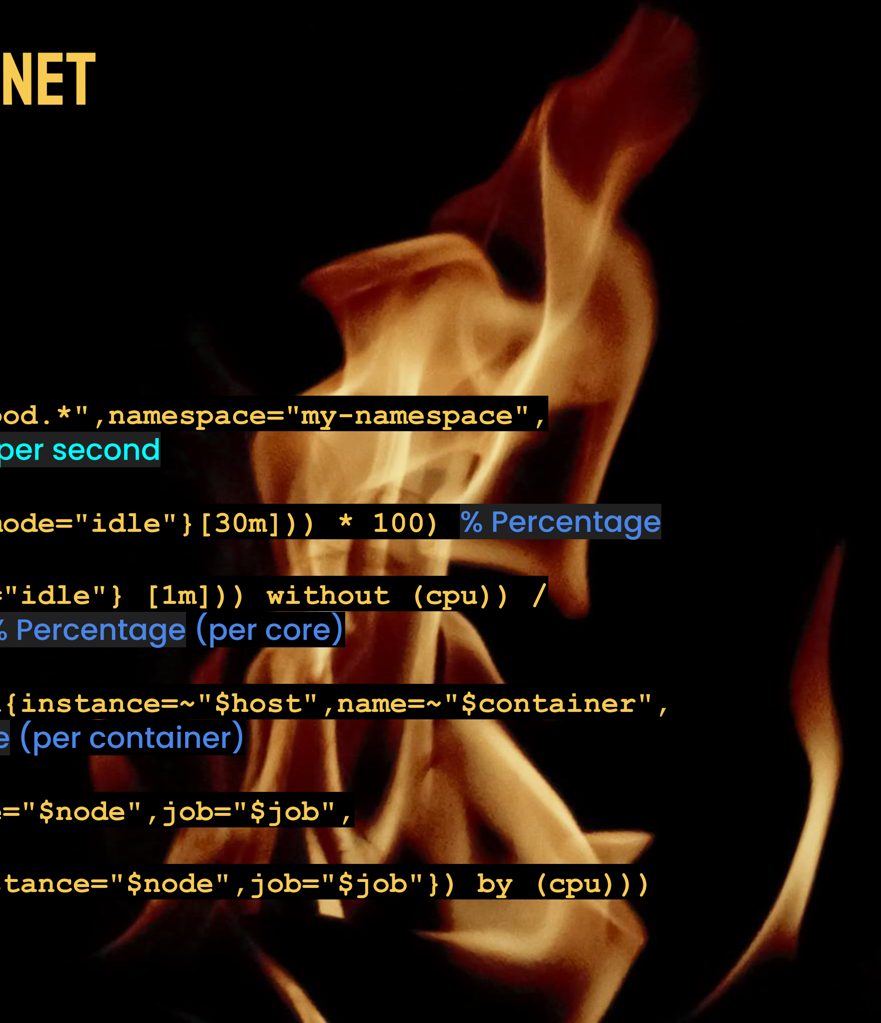
- `max by (container) (max by (pod, container, instance,environment,kubernetes_cluster) (rate(container_cpu_usage_seconds_total{container=~".*",instance=~".*",lens_service="$service_name",pod=~".*.*"}[5m])) / (max by (pod, container, instance,environment,kubernetes_cluster) (container_spec_cpu_shares{container=~".*",instance=~".*",pod=~".*.*"} / 1024) or max by (pod, container, instance,environment,kubernetes_cluster) (label_replace(kube_pod_container_resource_requests{container=~".*",node=~".*",pod=~".*.*",resource="cpu"}, "instance", "$1", "node", "(.*)"))))` % Percentage

8. CPU Usage as a Percentage of Quota

- `avg by (container) (max by (pod, container, instance,environment,kubernetes_cluster) (rate(container_cpu_usage_seconds_total{container=~".*",instance=~".*",lens_service="$service_name",pod=~".*.*"}[5m])) / (max by (pod, container, instance,environment,kubernetes_cluster) (container_spec_cpu_quota{container=~".*",instance=~".*"pod=~".*.*"} / 1e+06 or max by (pod, container, instance,environment,kubernetes_cluster) (label_replace(kube_pod_container_resource_limits{container=~".*",node=~".*",pod=~".*.*",resource="cpu"}, "instance", "$1", "node", "(.*)")))))` % Percentage



THE INTERNET



CADVISOR, NODE EXPORTER DASHBOARDS

```
9. rate
(container_cpu_usage_seconds_total{pod=~"my-pod.*",namespace="my-namespace",
container!=""}[$__rate_interval])
```

Core usage/per second

```
10. 100 - (avg(irate(node_cpu_seconds_total{mode="idle"}[30m])) * 100)
```

% Percentage

```
11. (sum(irate(node_cpu_seconds_total {mode!="idle"} [1m])) without (cpu)) /
count(node_cpu_seconds_total) without (cpu)
```

% Percentage (per core)

```
12.sum(rate(container_cpu_usage_seconds_total{instance=~"$host",name=~"$container",
name=~".+"}[5m])) by (name) *100
```

% Percentage (per container)

```
13. sum(irate(node_cpu_seconds_total{instance="$node",job="$job",
mode="system"}[$__rate_interval])) /
scalar(count(count(node_cpu_seconds_total{instance="$node",job="$job"}) by (cpu)))
```

% Percentage (per core)

THE WHEN

USAGE, PERCENTAGE, BYTES

Options Options Options
Container, KubeStateMetrics,
Node exporter Oh my

THE TYPES OF QUERIES

POD COUNT

:shrug:

%PERCENTAGE

Per core, per quota, per container...

OF CORES

Millicores or Cores

CORE USAGE PER SECOND

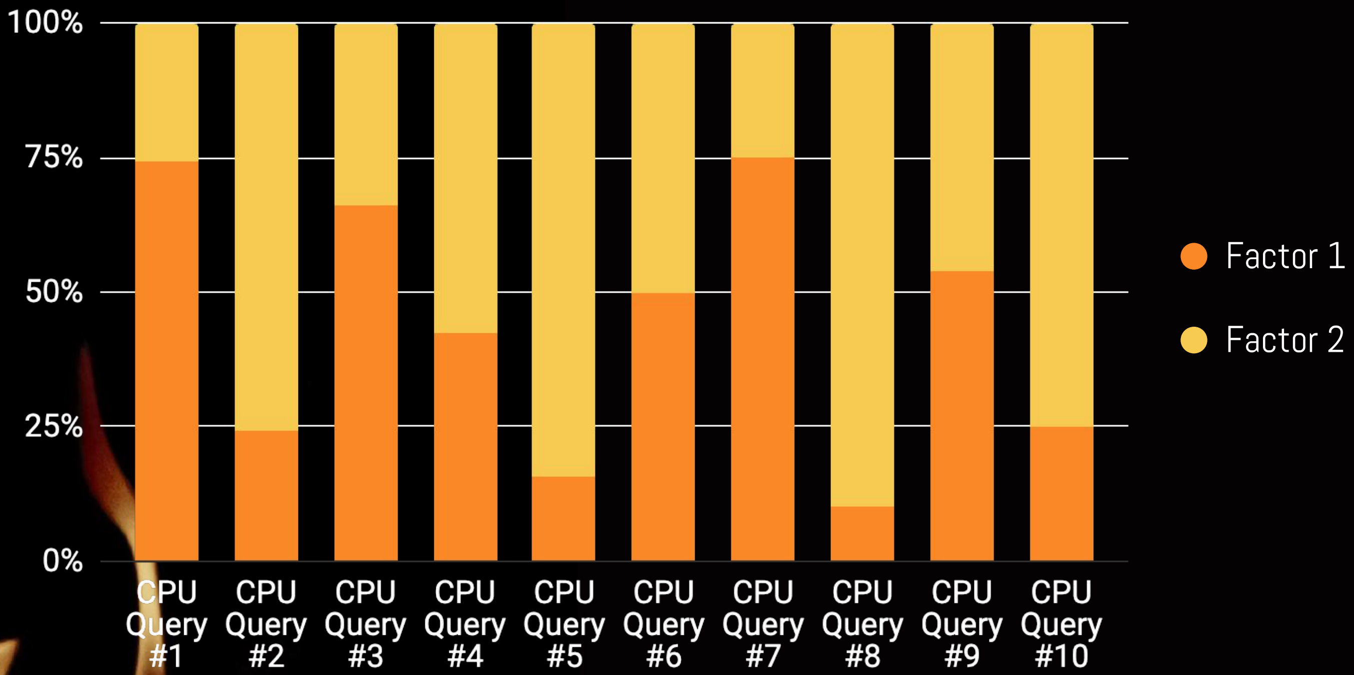
Usage

VOTE

Go to
https://app.sli.do/event/o3HQSXVyPCDmHDwUU8EdA2

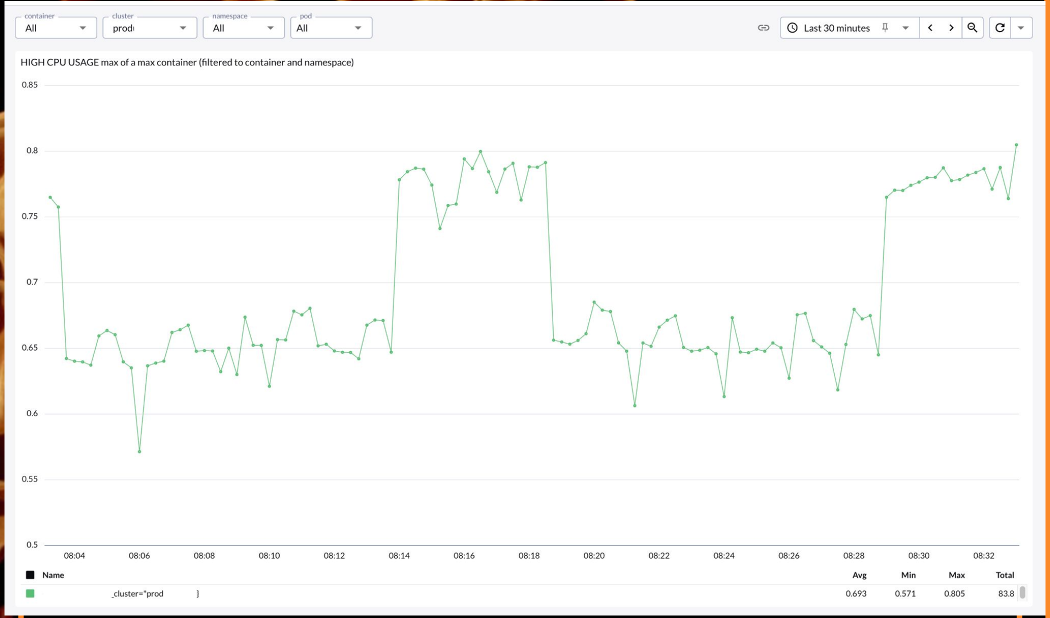
Go to slido.com #3245009

DUMMY RESULTS



REPEAT FOR MEMORY

```
 1.  max by (container) (max by (pod, container, instance,environment,cluster,namespace)
     (container_memory_working_set_bytes{container=~".*",instance=~".*",pod=~".*.*"}) / (max by (pod, container,
     instance,environment,cluster,namespace) (container_spec_memory_limit_bytes{container=~".*",instance=~".*",l,pod=~".*.*"}) or
     max by (pod, container, instance,environment,k8s_cluster,namespace)
     (label_replace(kube_pod_container_resource_limits{container=~".*",node=~".*",pod=~".*.*",resource="memory"}, "instance", "$1",
     "node", "(.*)"))))
 2.  max by (pod, container, instance,environment,cluster,namespace)
     (container_memory_working_set_bytes{container=~".*",instance=~".*",pod=~".*.*"})
 3.  sum by (pod, container, instance,environment,cluster,namespace)
     (container_memory_rss{container=~".*",instance=~".*",pod=~".*.*"})
 4.  node_memory_MemTotal_bytes{instance="10.1.1.37:9100",job="node-exporter"}
 5.  node_memory_MemTotal_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_MemFree_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     (node_memory_Cached_bytes{instance="10.1.1.37:9100",job="node-exporter"} +
     node_memory_Buffers_bytes{instance="10.1.1.37:9100",job="node-exporter"} +
     node_memory_SReclaimable_bytes{instance="10.1.1.37:9100",job="node-exporter"})
 6.  node_memory_Cached_bytes{instance="10.1.1.37:9100",job="node-exporter"} +
     node_memory_Buffers_bytes{instance="10.1.1.37:9100",job="node-exporter"} +
     node_memory_SReclaimable_bytes{instance="10.1.1.37:9100",job="node-exporter"}
 7.  node_memory_MemFree_bytes{instance="10.1.1.37:9100",job="node-exporter"}
 8.  (node_memory_SwapTotal_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_SwapFree_bytes{instance="10.1.1.37:9100",job="node-exporter"})
 9.  node_memory_MemTotal_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_MemFree_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_Buffers_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_Cached_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_Slab_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_PageTables_bytes{instance="10.1.1.37:9100",job="node-exporter"} -
     node_memory_SwapCached_bytes{instance="10.1.1.37:9100",job="node-exporter"}
10.  100*(sum by(kubernetes_cluster, pod, container, cloud_region) (container_memory_working_set_bytes{container!="POD",
     namespace=~"$namespace", environment=~"prod", container=~"app"}) / sum by(kubernetes_cluster, pod, container, cloud_region)
     (kube_pod_container_resource_requests{container!="",container!="POD",namespace=~"$namespace", resource="memory", unit="byte",
     environment=~"prod", container=~"app"}))
```



container: All · cluster: prod · namespace: All · pod: All

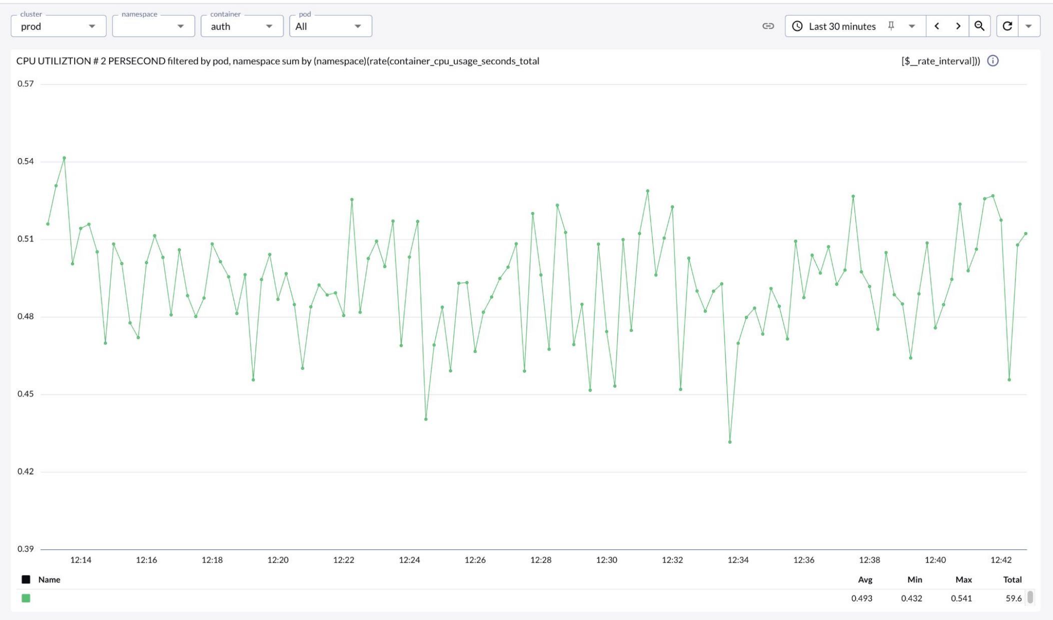
Last 30 minutes

HIGH CPU USAGE max of a max container (filtered to container and namespace)

Name	Avg	Min	Max	Total
🟩 _cluster="prod }	0.693	0.571	0.805	83.8

HIGH CPU USAGE

```
100
*
  max by (deployment, k8s_cluster, cloud_region) (
    label_replace(
      max by (pod, k8s_cluster, cloud_region) (
        rate(

container_cpu_usage_seconds_total{container=~"$container"
,image!="",namespace="$namespace"}[5m]
        )
      ),
      "deployment",
      "$1",
      "pod",
      "(.*)-.{8,10}-.{5}"
    )
  )
```



cluster: prod

namespace:

container: auth

pod: All

Last 30 minutes

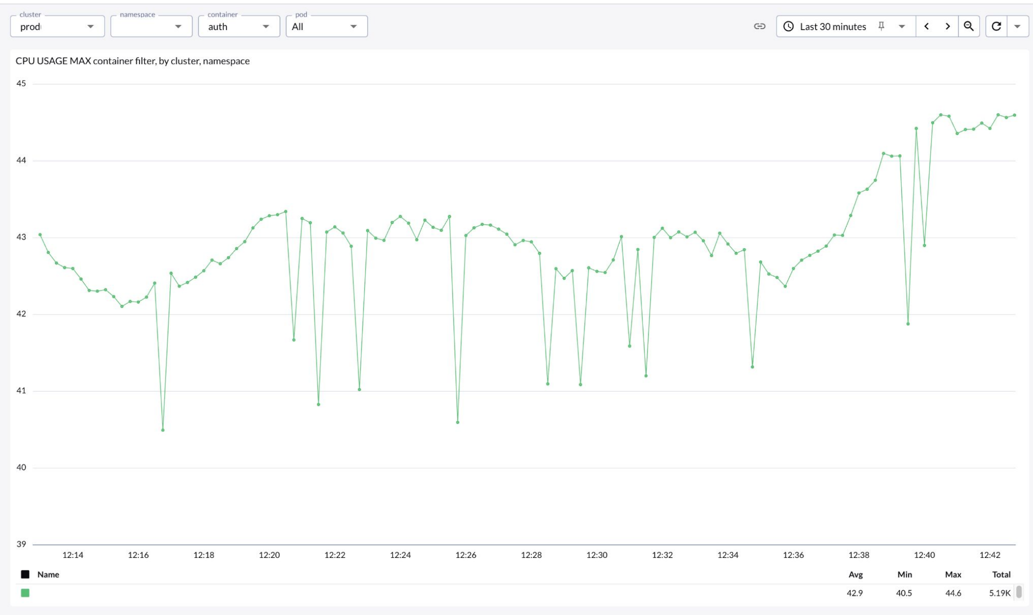
CPU UTILIZTION # 2 PERSECOND filtered by pod, namespace sum by (namespace)(rate(container_cpu_usage_seconds_total [$_rate_interval]))

Name	Avg	Min	Max	Total
🟩	0.493	0.432	0.541	59.6

CPU USAGE PER SECOND PER CORE ACROSS NAMESPACE

```
sum by (namespace) (

  rate(

container_cpu_usage_seconds_total{container=~"$container",
k8s_cluster=~"$cluster",namespace=~"$namespace"}[5m]

  )

)
```

HIGH CPU USAGE %



CPU USAGE MAX container filter, by cluster, namespace

Name	Avg	Min	Max	Total
🟩	42.9	40.5	44.6	5.19K

```
(
       max by (cloud_region, chronosphere_k8s_cluster,
namespace) (
          rate(

container_cpu_usage_seconds_total{chronosphere_k8s_cluste
r=~"$cluster",container=~"$container",namespace="$namespa
ce"}[5m]
          )
        )
     /
       max by (cloud_region, chronosphere_k8s_cluster,
namespace) (

kube_pod_container_resource_requests{chronosphere_k8s_clu
ster=~"$cluster",container=~"$container",namespace="$name
space",resource="cpu",unit="core"}
        )
   )
*
  100
```

THANKS!

Do you have any questions?
sarah.hudspeth@chronosphere.io
Come see us at Booth #H11
Check out the blogs on
chronosphere.io

Observability Day
NORTH AMERICA

CREDITS: This presentation template was created by Slidesgo, including icons by Flaticon, and infographics & images by Freepik.

chronosphere | kubecost | doit | Google Cloud

Cheers to Observability!

Thursday Nov 14, 2024 I 6:00-900 PM MT
The Rabbit Hole Lounge, Salt Lake City

SAVE MY SEAT

KubeCon | CloudNativeCon

Find Chronosphere @ Booth #H11

RESOURCES

- Grafana Open Source Dashboards
  - https://grafana.com/grafana/dashboards/1860-node-exporter-full/
  - https://grafana.com/grafana/dashboards/14282-cadvisor-exporter/
- TenPlus CPU and Memory Queries (Perses Open Source Dashboard)
  - https://demo.perses.dev/projects/cpu_and_memory_usage_queries___kubecon2024
  - PLEASE GO ADD MORE QUERIES – MAKE YOUR OWN DASHBOARD
- Stack Overflow Article
  - https://stackoverflow.com/questions/34923788/prometheus-convert-cpu-user-seconds-to-cpu-usage