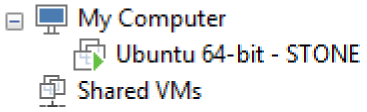
Monitoração de Cluster Kubernetes utilizando Prometheus e Grafana

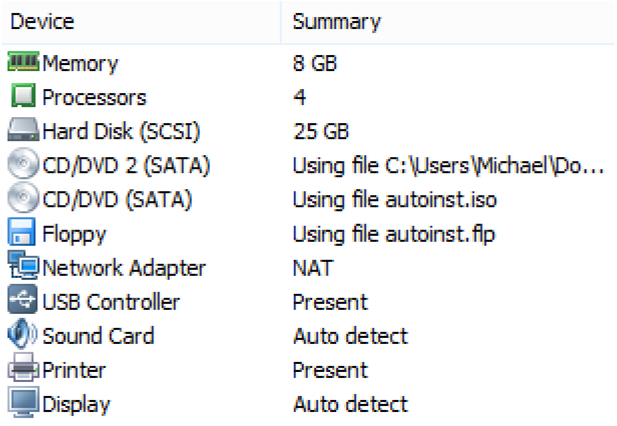
Este documento cobre todos os passos utilizados para implementar uma solução de monitoração para o cluster de Kubernetes

Premissas

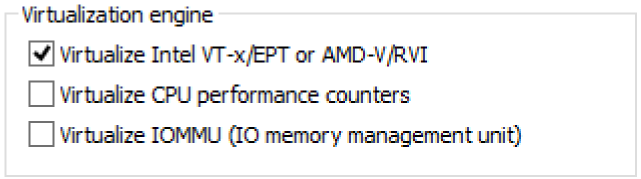
Para demonstrar essa solução, foi necessário a utilização de um servidor de laboratório. Neste caso, foi utilizado o Ubuntu-20.04.1-live-server-amd64. O mesmo foi instalado em uma Virtual Machine, usando VMware® Workstation 14 Pro, conforme as configurações abaixo:



- My Computer
  - Ubuntu 64-bit - STONE
- Shared VMs



Device	Summary
Memory	8 GB
Processors	4
Hard Disk (SCSI)	25 GB
CD/DVD 2 (SATA)	Using file C:\Users\Michael\Do...
CD/DVD (SATA)	Using file autoinst.iso
Floppy	Using file autoinst.flp
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect



Virtualization engine

☑ Virtualize Intel VT-x/EPT or AMD-V/RVI

□ Virtualize CPU performance counters

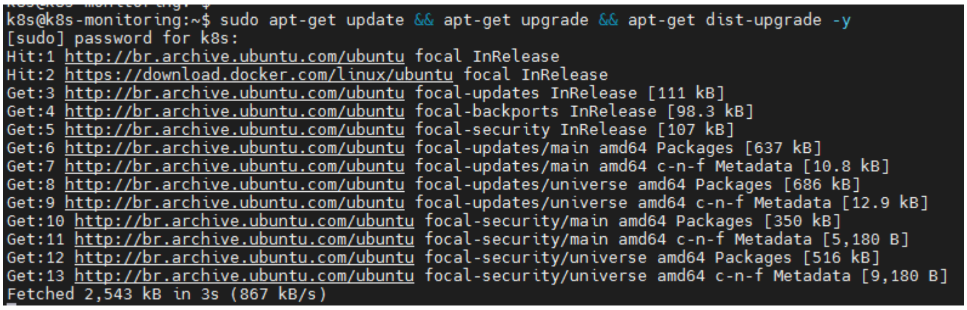
□ Virtualize IOMMU (IO memory management unit)

Implantação

Os seguintes passos foram executados para garantir a implantação dessa solução.

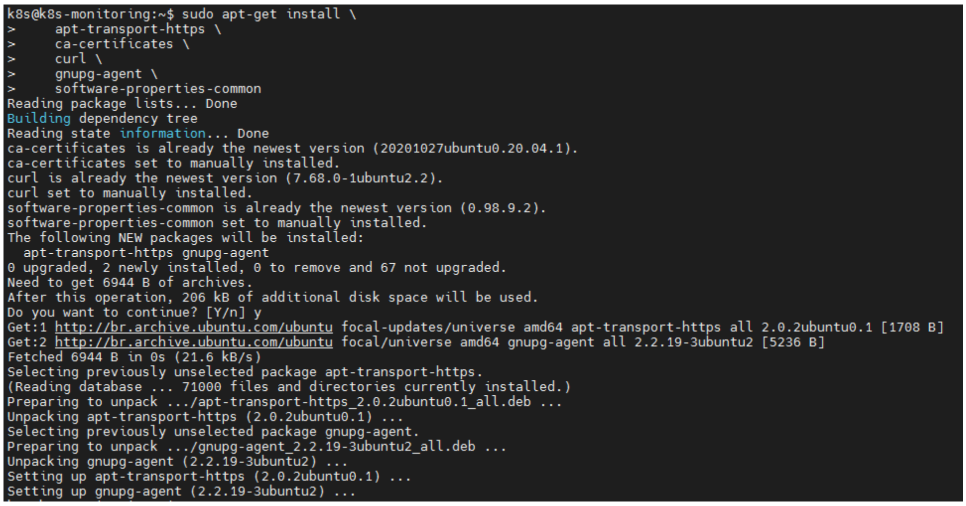
1. Docker Engine

1.1 O comando abaixo foi utilizado para atualização da lista de repositórios e pacotes instalados no sistema:



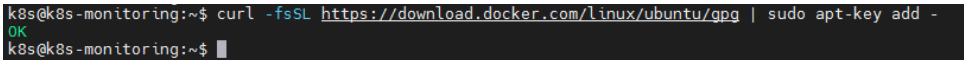
```
k8s@k8s-monitoring:~$ sudo apt-get update && apt-get upgrade && apt-get dist-upgrade -y
[sudo] password for k8s:
Hit:1 http://br.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 https://download.docker.com/linux/ubuntu focal InRelease
Get:3 http://br.archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Get:4 http://br.archive.ubuntu.com/ubuntu focal-backports InRelease [98.3 kB]
Get:5 http://br.archive.ubuntu.com/ubuntu focal-security InRelease [107 kB]
Get:6 http://br.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [637 kB]
Get:7 http://br.archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [10.8 kB]
Get:8 http://br.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [686 kB]
Get:9 http://br.archive.ubuntu.com/ubuntu focal-updates/universe amd64 c-n-f Metadata [12.9 kB]
Get:10 http://br.archive.ubuntu.com/ubuntu focal-security/main amd64 Packages [350 kB]
Get:11 http://br.archive.ubuntu.com/ubuntu focal-security/main amd64 c-n-f Metadata [5,180 B]
Get:12 http://br.archive.ubuntu.com/ubuntu focal-security/universe amd64 Packages [516 kB]
Get:13 http://br.archive.ubuntu.com/ubuntu focal-security/universe amd64 c-n-f Metadata [9,180 B]
Fetched 2,543 kB in 3s (867 kB/s)
```

1.2 Após atualizar todo o sistema, realizei a instalação de alguns pacotes solicitados na documentação oficial para que o Docker funcione corretamente:



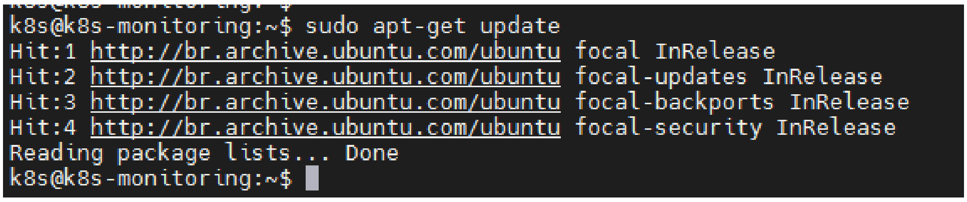
```
k8s@k8s-monitoring:~$ sudo apt-get install \
>     apt-transport-https \
>     ca-certificates \
>     curl \
>     gnupg-agent \
>     software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
ca-certificates is already the newest version (20201027ubuntu0.20.04.1).
ca-certificates set to manually installed.
curl is already the newest version (7.68.0-1ubuntu2.2).
curl set to manually installed.
software-properties-common is already the newest version (0.98.9.2).
software-properties-common set to manually installed.
The following NEW packages will be installed:
  apt-transport-https gnupg-agent
0 upgraded, 2 newly installed, 0 to remove and 67 not upgraded.
Need to get 6944 B of archives.
After this operation, 206 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://br.archive.ubuntu.com/ubuntu focal-updates/universe amd64 apt-transport-https all 2.0.2ubuntu0.1 [1708 B]
Get:2 http://br.archive.ubuntu.com/ubuntu focal/universe amd64 gnupg-agent all 2.2.19-3ubuntu2 [5236 B]
Fetched 6944 B in 0s (21.6 kB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 71000 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.0.2ubuntu0.1_all.deb ...
Unpacking apt-transport-https (2.0.2ubuntu0.1) ...
Selecting previously unselected package gnupg-agent.
Preparing to unpack .../gnupg-agent_2.2.19-3ubuntu2_all.deb ...
Unpacking gnupg-agent (2.2.19-3ubuntu2) ...
Setting up apt-transport-https (2.0.2ubuntu0.1) ...
Setting up gnupg-agent (2.2.19-3ubuntu2) ...
```

1.3 Também adicionei a GPG Key do docker para mapeamento dos repositórios oficiais:



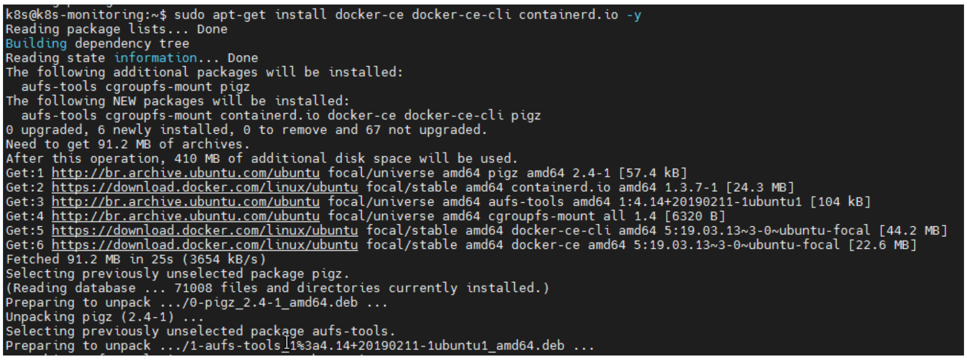
```
k8s@k8s-monitoring:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
OK
k8s@k8s-monitoring:~$
```

1.4 Atualizei novamente a lista de repositórios para que o repositório do docker ficasse disponível para uso:



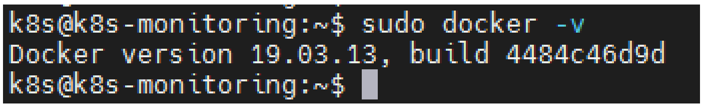
```
k8s@k8s-monitoring:~$ sudo apt-get update
Hit:1 http://br.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://br.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://br.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://br.archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
k8s@k8s-monitoring:~$
```

1.5 Instalação do Docker e suas dependências:

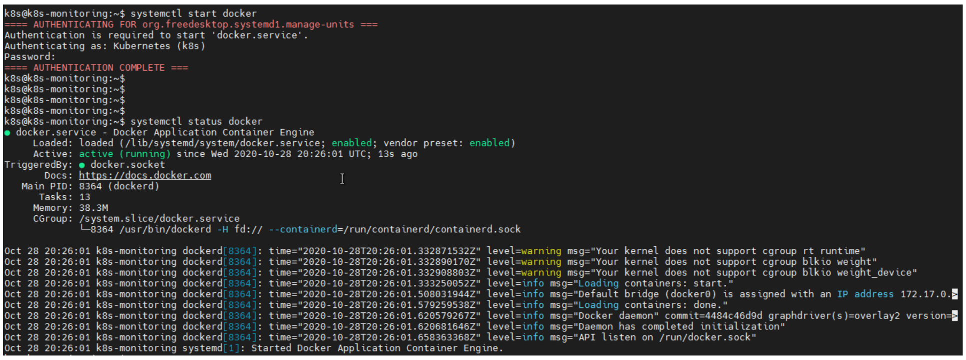


```
k8s@k8s-monitoring:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  aufs-tools cgroupfs-mount pigz
The following NEW packages will be installed:
  aufs-tools cgroupfs-mount containerd.io docker-ce docker-ce-cli pigz
0 upgraded, 6 newly installed, 0 to remove and 67 not upgraded.
Need to get 91.2 MB of archives.
After this operation, 410 MB of additional disk space will be used.
Get:1 http://br.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57.4 kB]
Get:2 https://download.docker.com/linux/ubuntu focal/stable amd64 containerd.io amd64 1.3.7-1 [24.3 MB]
Get:3 http://br.archive.ubuntu.com/ubuntu focal/universe amd64 aufs-tools amd64 1:4.14+20190211-1ubuntu1 [104 kB]
Get:4 http://br.archive.ubuntu.com/ubuntu focal/universe amd64 cgroupfs-mount all 1.4 [6320 B]
Get:5 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce-cli amd64 5:19.03.13~3-0~ubuntu-focal [44.2 MB]
Get:6 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce amd64 5:19.03.13~3-0~ubuntu-focal [22.6 MB]
Fetched 91.2 MB in 25s (3654 kB/s)
Selecting previously unselected package pigz.
(Reading database ... 71008 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.4-1_amd64.deb ...
Unpacking pigz (2.4-1) ...
Selecting previously unselected package aufs-tools.
Preparing to unpack .../1-aufs-tools_1%3a4.14+20190211-1ubuntu1_amd64.deb ...
```

1.6 Após instalação, verifiquei se o Docker foi instalado corretamente com os comandos abaixo:



```
k8s@k8s-monitoring:~$ sudo docker -v
Docker version 19.03.13, build 4484c46d9d
k8s@k8s-monitoring:~$
```

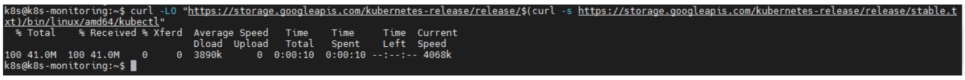


```
k8s@k8s-monitoring:~$ systemctl start docker
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to start 'docker.service'.
Authenticating as: Kubernetes (k8s)
Password:
==== AUTHENTICATION COMPLETE ===
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$ systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2020-10-28 20:26:01 UTC; 13s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 8364 (dockerd)
      Tasks: 13
     Memory: 38.3M
     CGroup: /system.slice/docker.service
             └─8364 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.332871532Z" level=warning msg="Your kernel does not support cgroup rt runtime"
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.332890170Z" level=warning msg="Your kernel does not support cgroup blkio weight"
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.332908803Z" level=warning msg="Your kernel does not support cgroup blkio weight_device"
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.333250052Z" level=info msg="Loading containers: start."
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.508031944Z" level=info msg="Default bridge (docker0) is assigned with an IP address 172.17.0.>
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.579259538Z" level=info msg="Loading containers: done."
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.620579267Z" level=info msg="Docker daemon" commit=4484c46d9d graphdriver(s)=overlay2 version=>
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.620681646Z" level=info msg="Daemon has completed initialization"
Oct 28 20:26:01 k8s-monitoring dockerd[8364]: time="2020-10-28T20:26:01.658363368Z" level=info msg="API listen on /run/docker.sock"
Oct 28 20:26:01 k8s-monitoring systemd[1]: Started Docker Application Container Engine.
```

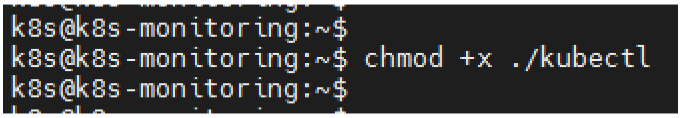
2. KubeCTL

2.1 Realizei o download do KubeCTL utilizando o comando:



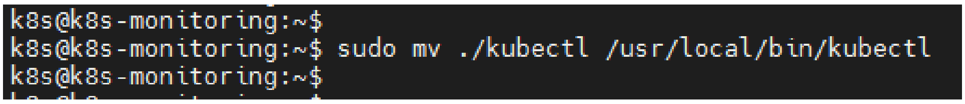
```
k8s@k8s-monitoring:~$ curl -LO "https://storage.googleapis.com/kubernetes-release/release/$(curl -s https://storage.googleapis.com/kubernetes-release/release/stable.t
xt)/bin/linux/amd64/kubectl"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 41.0M  100 41.0M    0     0  3890k      0  0:00:10  0:00:10 --:--:-- 4068k
k8s@k8s-monitoring:~$ 
```

2.2 Adicionei a permissão de execução no arquivo binário:



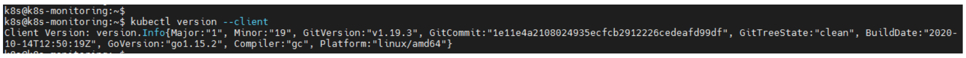
```
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$ chmod +x ./kubectl
k8s@k8s-monitoring:~$
```

2.3 Movi o binário para o diretório abaixo visando garantir que o comando *kubectl* possa ser executado a partir de qualquer diretório:



```
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$ sudo mv ./kubectl /usr/local/bin/kubectl
k8s@k8s-monitoring:~$
```

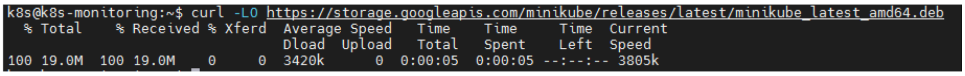
2.4 Executei o seguinte comando para confirmar se as configurações anteriores foram realizadas com sucesso:



```
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$ kubectl version --client
Client Version: version.Info{Major:"1", Minor:"19", GitVersion:"v1.19.3", GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df", GitTreeState:"clean", BuildDate:"2020-
10-14T12:50:19Z", GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

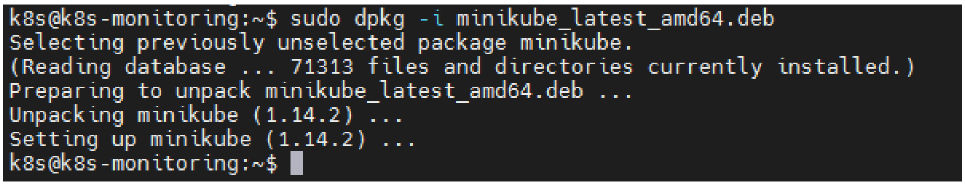
3. MiniKube

3.1 Realizei o download do MiniKube:



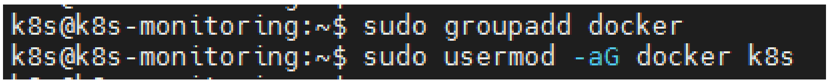
```
k8s@k8s-monitoring:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube_latest_amd64.deb
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 19.0M  100 19.0M    0     0  3420k      0  0:00:05  0:00:05 --:--:-- 3805k
```

3.2 Após o download, realizei a instalação do MiniKube:



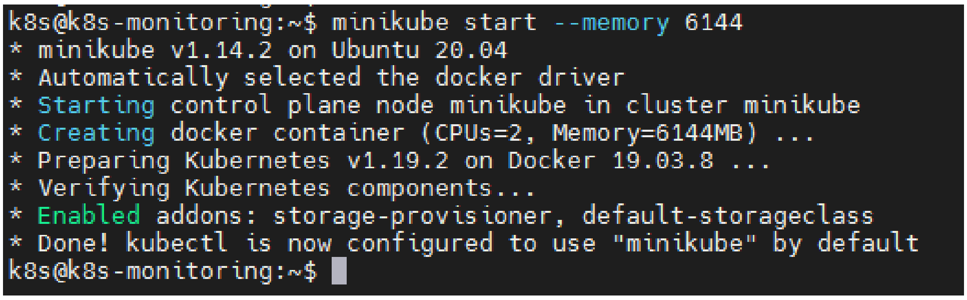
```
k8s@k8s-monitoring:~$ sudo dpkg -i minikube_latest_amd64.deb
Selecting previously unselected package minikube.
(Reading database ... 71313 files and directories currently installed.)
Preparing to unpack minikube_latest_amd64.deb ...
Unpacking minikube (1.14.2) ...
Setting up minikube (1.14.2) ...
k8s@k8s-monitoring:~$
```

3.3 Para iniciar o MiniKube corretamente, foi necessário criar um grupo chamado *docker* e inserir o meu usuário linux *k8s* no mesmo:



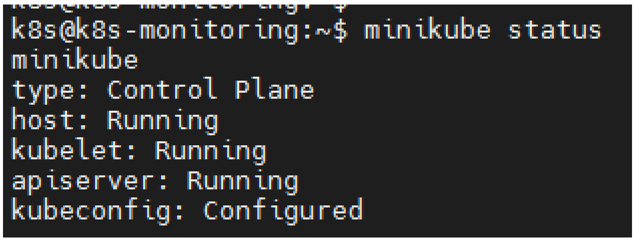
```
k8s@k8s-monitoring:~$ sudo groupadd docker
k8s@k8s-monitoring:~$ sudo usermod -aG docker k8s
```

3.4 Iniciei o MiniKube:

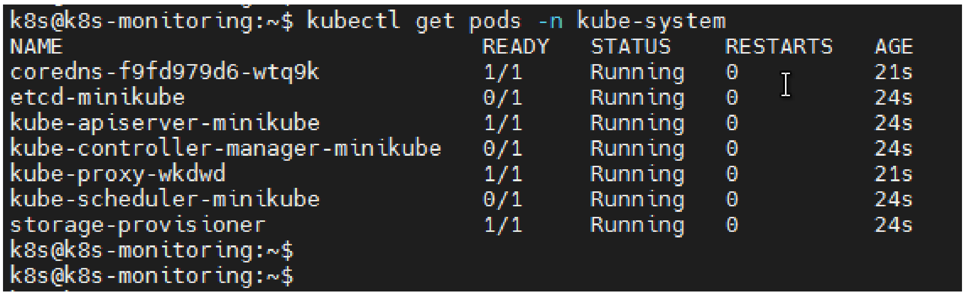


```
k8s@k8s-monitoring:~$ minikube start --memory 6144
* minikube v1.14.2 on Ubuntu 20.04
* Automatically selected the docker driver
* Starting control plane node minikube in cluster minikube
* Creating docker container (CPUs=2, Memory=6144MB) ...
* Preparing Kubernetes v1.19.2 on Docker 19.03.8 ...
* Verifying Kubernetes components...
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" by default
k8s@k8s-monitoring:~$
```

3.5. Para validar a instalação do MiniKube, executei alguns comandos:



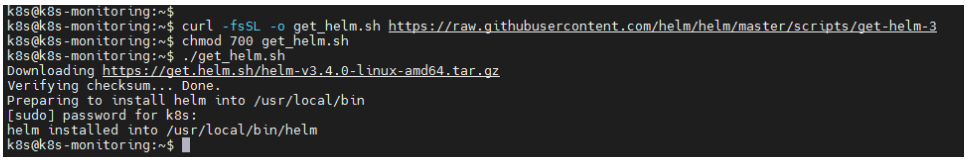
```
k8s@k8s-monitoring:~$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
```



```
k8s@k8s-monitoring:~$ kubectl get pods -n kube-system
NAME                               READY   STATUS    RESTARTS   AGE
coredns-f9fd979d6-wtq9k            1/1     Running   0          21s
etcd-minikube                      0/1     Running   0          24s
kube-apiserver-minikube            1/1     Running   0          24s
kube-controller-manager-minikube   0/1     Running   0          24s
kube-proxy-wkdwd                   1/1     Running   0          21s
kube-scheduler-minikube            0/1     Running   0          24s
storage-provisioner                1/1     Running   0          24s
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$
```

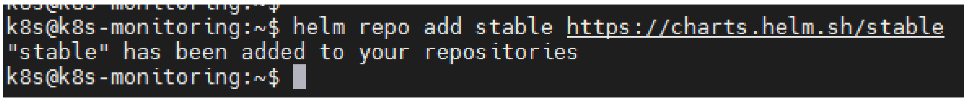
4. Prometheus e Grafana

4.1 Realizei o download do script de instalação do Helm, adicionei as permissões no mesmo e o executei:



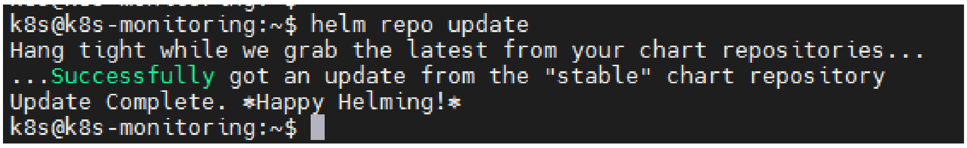
```
k8s@k8s-monitoring:~$
k8s@k8s-monitoring:~$ curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
k8s@k8s-monitoring:~$ chmod 700 get_helm.sh
k8s@k8s-monitoring:~$ ./get_helm.sh
Downloading https://get.helm.sh/helm-v3.4.0-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
[sudo] password for k8s:
helm installed into /usr/local/bin/helm
k8s@k8s-monitoring:~$
```

4.2 Adicionei ao meu repositório local o repositório de charts do Helm.



```
k8s@k8s-monitoring:~$ helm repo add stable https://charts.helm.sh/stable
"stable" has been added to your repositories
k8s@k8s-monitoring:~$
```

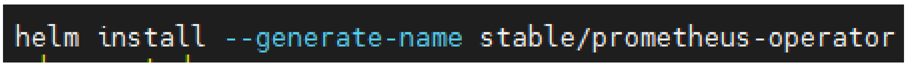
4.3 Atualizei a minha lista de repositórios locais:



```
k8s@k8s-monitoring:~$ helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "stable" chart repository
Update Complete. *Happy Helming!*
k8s@k8s-monitoring:~$
```

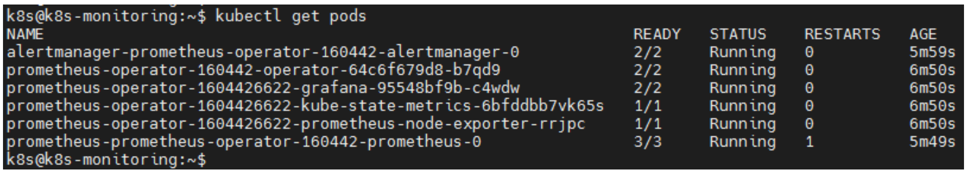
4.4 Com o comando abaixo, fiz a instalação do Prometheus Operator, suíte de monitoração completa para o Kubernetes, contendo os seguintes pacotes:

- Prometheus
- Grafana
- Node Exporter
- Kube State Metrics
- Alert Manager

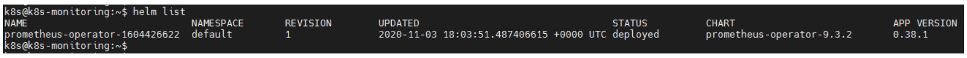


```
helm install --generate-name stable/prometheus-operator
```

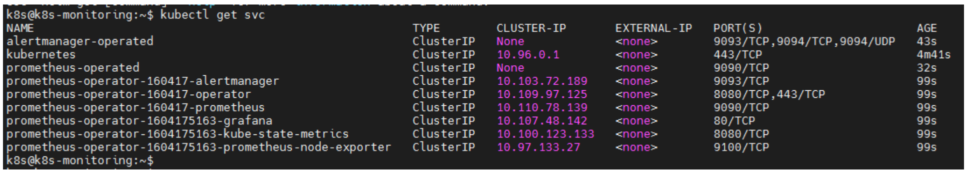
4.5 Verifiquei o status da instalação do Prometheus Operator:



```
k8s@k8s-monitoring:~$ kubectl get pods
NAME                                                        READY   STATUS    RESTARTS   AGE
alertmanager-prometheus-operator-160442-alertmanager-0      2/2     Running   0          5m59s
prometheus-operator-160442-operator-64c6f679d8-b7qd9        2/2     Running   0          6m50s
prometheus-operator-1604426622-grafana-95548bf9b-c4wdw      2/2     Running   0          6m50s
prometheus-operator-1604426622-kube-state-metrics-6bfddbb7vk65s   1/1   Running   0    6m50s
prometheus-operator-1604426622-prometheus-node-exporter-rrjpc     1/1   Running   0    6m50s
prometheus-prometheus-operator-160442-prometheus-0          3/3     Running   1          5m49s
k8s@k8s-monitoring:~$
```

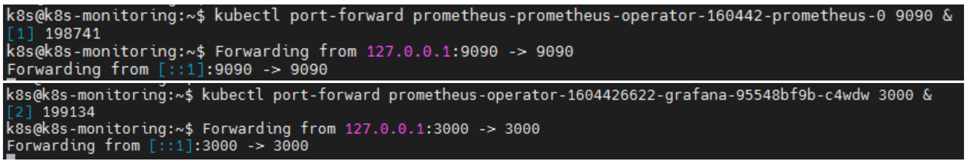


```
k8s@k8s-monitoring:~$ helm list
NAME                            NAMESPACE       REVISION        UPDATED                                 STATUS          CHART                           APP VERSION
prometheus-operator-1604426622  default         1               2020-11-03 18:03:51.487406615 +0000 UTC deployed        prometheus-operator-9.3.2       0.38.1
k8s@k8s-monitoring:~$
```



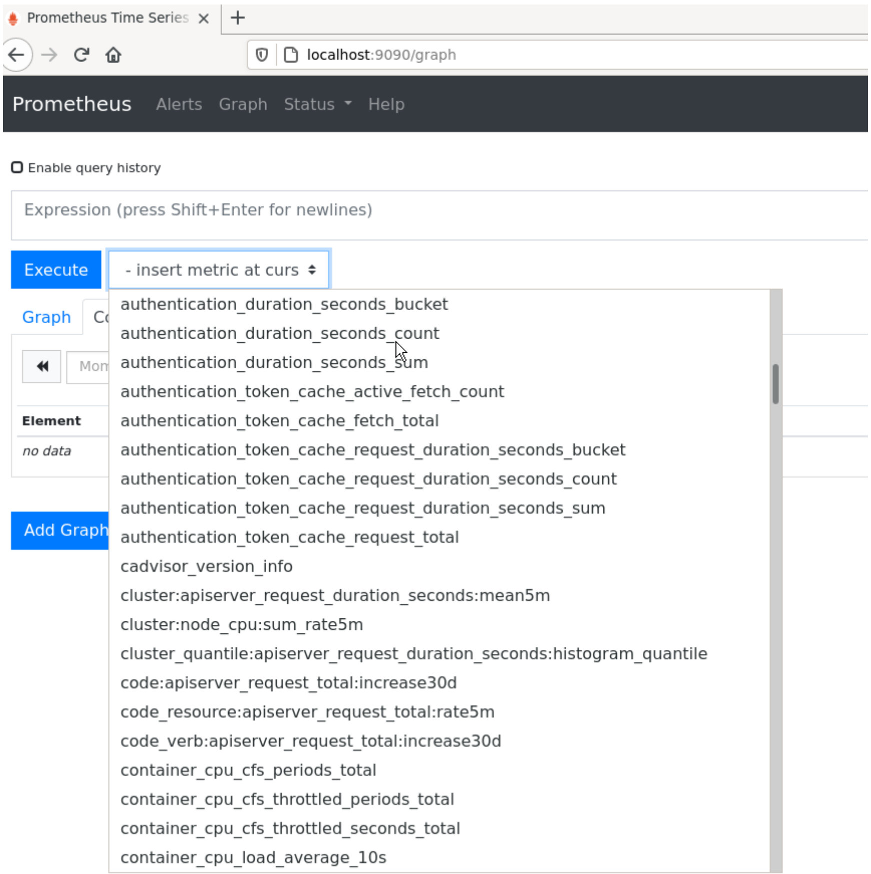
```
k8s@k8s-monitoring:~$ kubectl get svc
NAME                                                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                      AGE
alertmanager-operated                                ClusterIP   None             <none>        9093/TCP,9094/TCP,9094/UDP   43s
kubernetes                                           ClusterIP   10.96.0.1        <none>        443/TCP                      4m41s
prometheus-operated                                  ClusterIP   None             <none>        9090/TCP                     32s
prometheus-operator-160417-alertmanager              ClusterIP   10.103.72.189    <none>        9093/TCP                     99s
prometheus-operator-160417-operator                  ClusterIP   10.109.97.125    <none>        8080/TCP,443/TCP             99s
prometheus-operator-160417-prometheus                ClusterIP   10.110.78.139    <none>        9090/TCP                     99s
prometheus-operator-1604175163-grafana               ClusterIP   10.107.48.142    <none>        80/TCP                       99s
prometheus-operator-1604175163-kube-state-metrics    ClusterIP   10.100.123.133   <none>        8080/TCP                     99s
prometheus-operator-1604175163-prometheus-node-exporter   ClusterIP   10.97.133.27   <none>     9100/TCP                     99s
k8s@k8s-monitoring:~$
```

4.6 Por fim, liberei o acesso às portas correspondentes dos serviços *Prometheus* e *Grafana:*

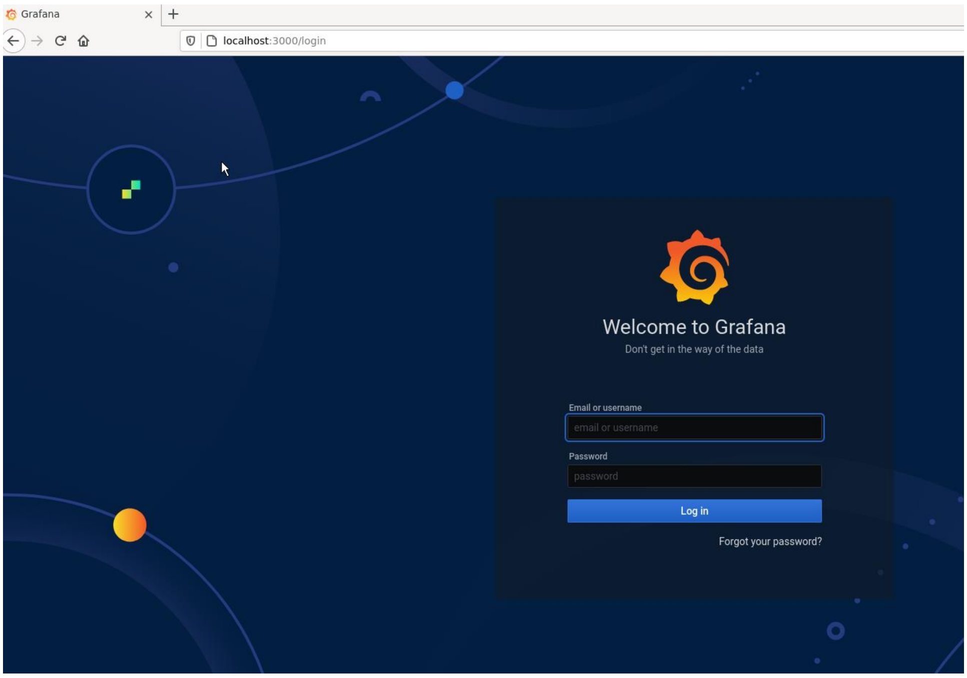


```
k8s@k8s-monitoring:~$ kubectl port-forward prometheus-prometheus-operator-160442-prometheus-0 9090 &
[1] 198741
k8s@k8s-monitoring:~$ Forwarding from 127.0.0.1:9090 -> 9090
Forwarding from [::1]:9090 -> 9090
k8s@k8s-monitoring:~$ kubectl port-forward prometheus-operator-1604426622-grafana-95548bf9b-c4wdw 3000 &
[2] 199134
k8s@k8s-monitoring:~$ Forwarding from 127.0.0.1:3000 -> 3000
Forwarding from [::1]:3000 -> 3000
```

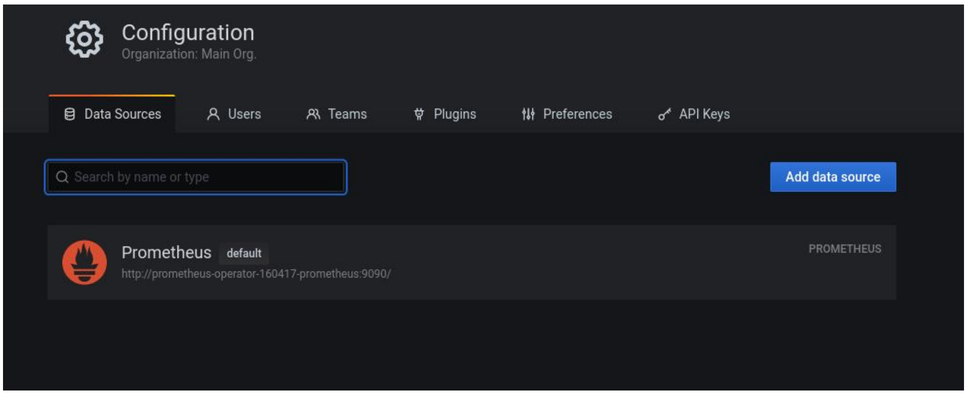
4.7 Para validar os serviços, acessei os mesmos pelo browser utilizando suas respectivas portas:



authentication_duration_seconds_bucket
authentication_duration_seconds_count
authentication_duration_seconds_sum
authentication_token_cache_active_fetch_count
authentication_token_cache_fetch_total
authentication_token_cache_request_duration_seconds_bucket
authentication_token_cache_request_duration_seconds_count
authentication_token_cache_request_duration_seconds_sum
authentication_token_cache_request_total
cadvisor_version_info
cluster:apiserver_request_duration_seconds:mean5m
cluster:node_cpu:sum_rate5m
cluster_quantile:apiserver_request_duration_seconds:histogram_quantile
code:apiserver_request_total:increase30d
code_resource:apiserver_request_total:rate5m
code_verb:apiserver_request_total:increase30d
container_cpu_cfs_periods_total
container_cpu_cfs_throttled_periods_total
container_cpu_cfs_throttled_seconds_total
container_cpu_load_average_10s



4.8 Após digitar usuário e senha, acessei a aba de *Data Sources* a fim de verificar se o Prometheus estava associado como um Data Source válido no Grafana:



Prometheus default
http://prometheus-operator-160417-prometheus:9090/

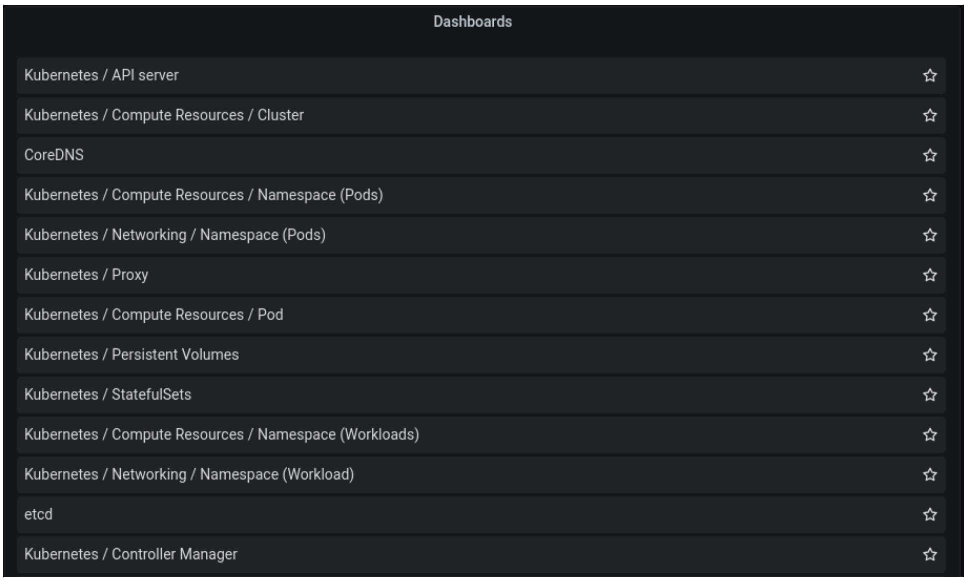
Validação

A solução proposta cobre parcialmente o item "Entregável 4", conforme detalhado abaixo:

- Monitorar o cluster de Kubernetes, ~~definindo alertas importantes.~~
- ~~Monitorar a aplicação dentro do cluster de Kubernetes~~.
- Criar dashboards que mostrem o estado da ~~aplicação~~ e do cluster.

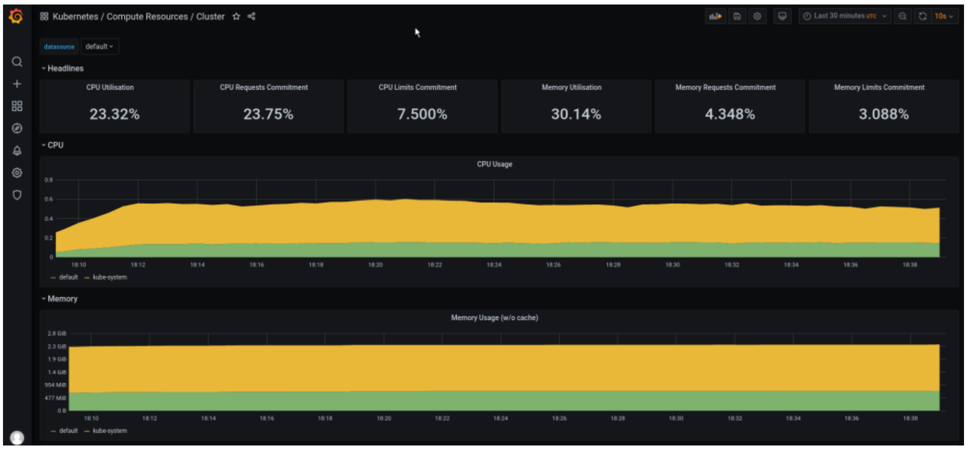
Referência: https://gist.github.com/biancarosa/1f91e4c2bdce104d507ed65d00b4b009

Para validar a solução, aguardei o tempo dos gráficos serem populados, acessei a aba *Dashboards* e depois cliquei em alguns dashboards relacionados ao Kubernetes:



Dashboards

- Kubernetes / API server ☆
- Kubernetes / Compute Resources / Cluster ☆
- CoreDNS ☆
- Kubernetes / Compute Resources / Namespace (Pods) ☆
- Kubernetes / Networking / Namespace (Pods) ☆
- Kubernetes / Proxy ☆
- Kubernetes / Compute Resources / Pod ☆
- Kubernetes / Persistent Volumes ☆
- Kubernetes / StatefulSets ☆
- Kubernetes / Compute Resources / Namespace (Workloads) ☆
- Kubernetes / Networking / Namespace (Workload) ☆
- etcd ☆
- Kubernetes / Controller Manager ☆

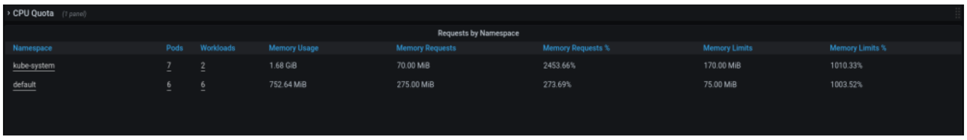


Kubernetes / Compute Resources / Cluster ☆

datasource default

Headlines

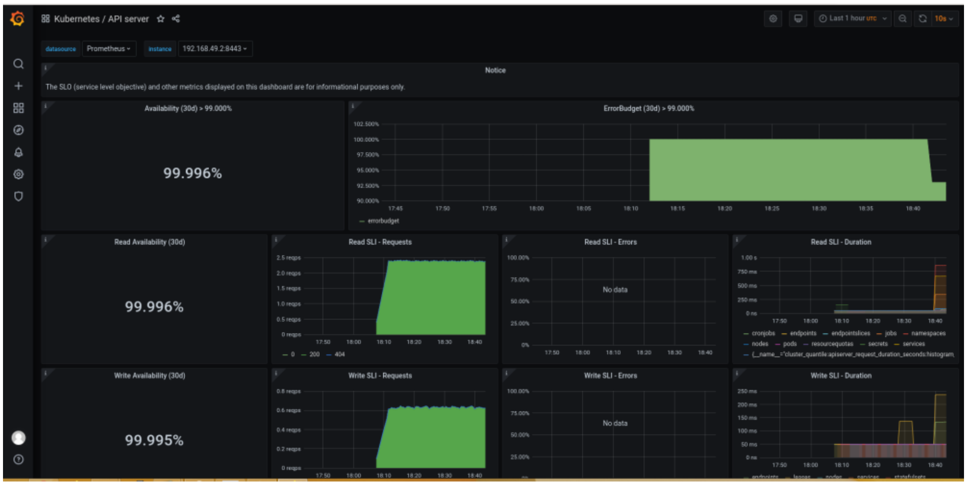
CPU Utilisation	CPU Requests Commitment	CPU Limits Commitment	Memory Utilisation	Memory Requests Commitment	Memory Limits Commitment
23.32%	23.75%	7.500%	30.14%	4.348%	3.088%



› CPU Quota (1 panel)

Requests by Namespace

Namespace	Pods	Workloads	Memory Usage	Memory Requests	Memory Requests %	Memory Limits	Memory Limits %
kube-system	7	2	1.68 GiB	70.00 MiB	2453.66%	170.00 MiB	1010.33%
default	6	6	752.64 MiB	275.00 MiB	273.69%	75.00 MiB	1003.52%



Kubernetes / API server

datasource Prometheus · instance 192.168.49.2:8443

Notice

The SLO (service level objective) and other metrics displayed on this dashboard are for informational purposes only.

Availability (30d) > 99.000%

99.996%

ErrorBudget (30d) > 99.000%

Read Availability (30d)

99.996%

Read SLI - Requests

Read SLI - Errors

No data

Read SLI - Duration

Write Availability (30d)

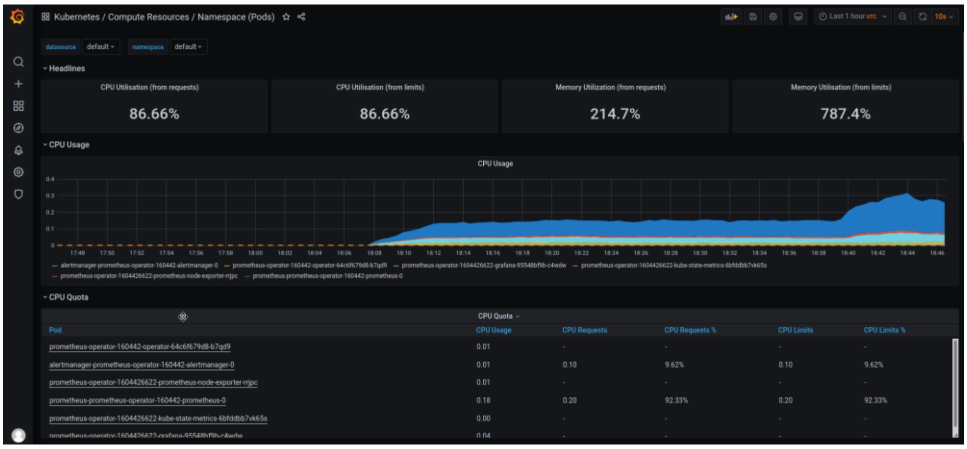
99.995%

Write SLI - Requests

Write SLI - Errors

No data

Write SLI - Duration



Kubernetes / Compute Resources / Namespace (Pods)

datasource default · namespace default

Headlines

CPU Utilisation (from requests)	CPU Utilisation (from limits)	Memory Utilization (from requests)	Memory Utilisation (from limits)
86.66%	86.66%	214.7%	787.4%

CPU Usage

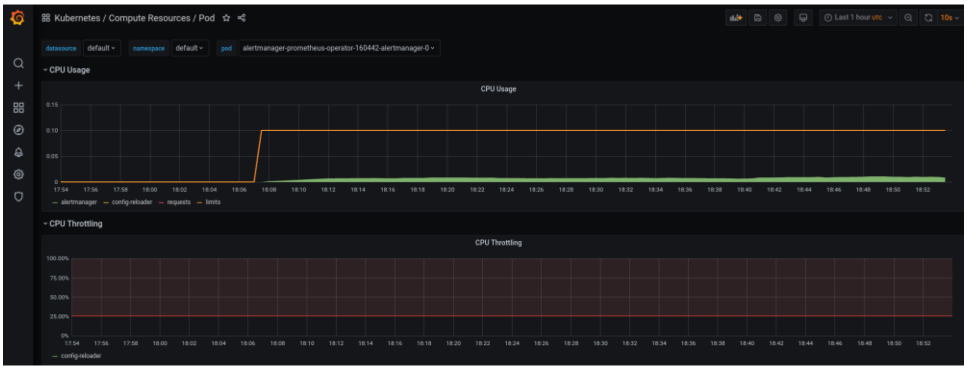
CPU Usage

Last 1 hour UTC
10s
0.4
0.3
0.2
0.1
0
17:48
17:50
17:52
17:54
17:56
17:58
18:00
18:02
18:04
18:06
18:08
18:10
18:12
18:14
18:16
18:18
18:20
18:22
18:24
18:26
18:28
18:30
18:32
18:34
18:36
18:38
18:40
18:42
18:44
18:46
alertmanager-prometheus-operator-160442-alertmanager-0
prometheus-operator-160442-operator-64c6f679d8-b7qd9
prometheus-operator-1604426622-grafana-95548bf9b-c4wdw
prometheus-operator-1604426622-kube-state-metrics-6bfddbb7vk65s
prometheus-operator-1604426622-prometheus-node-exporter-rrjpc
prometheus-prometheus-operator-160442-prometheus-0

CPU Quota

CPU Quota

Pod	CPU Usage	CPU Requests	CPU Requests %	CPU Limits	CPU Limits %
prometheus-operator-160442-operator-64c6f679d8-b7qd9	0.01	-	-	-	-
alertmanager-prometheus-operator-160442-alertmanager-0	0.01	0.10	9.62%	0.10	9.62%
prometheus-operator-1604426622-prometheus-node-exporter-rrjpc	0.01	-	-	-	-
prometheus-prometheus-operator-160442-prometheus-0	0.18	0.20	92.33%	0.20	92.33%
prometheus-operator-1604426622-kube-state-metrics-6bfddbb7vk65s	0.00	-	-	-	-
prometheus-operator-1604426622-grafana-95548bf9b-c4wdw	0.04	-	-	-	-



Kubernetes / Compute Resources / Pod

datasource default · namespace default · pod alertmanager-prometheus-operator-160442-alertmanager-0

CPU Usage

CPU Usage

CPU Throttling

CPU Throttling

Referências

Seguem abaixo algumas referências que utilizei para fazer a instalação e o troubleshooting durante todo o processo:

- https://docs.docker.com/engine/install/ubuntu/
- https://minikube.sigs.k8s.io/docs/start/
- https://github.com/helm/charts/tree/master/stable/prometheus-operator#configuration
- https://github.com/helm/helm/releases
- https://github.com/kubernetes/kube-state-metrics
- https://medium.com/htc-research-engineering-blog/monitoring-kubernetes-clusters-with-grafana-e2a413febefd
- https://stackoverflow.com/questions/59480373/validationerror-missing-required-field-selector-in-io-k8s-api-v1-deploymentsp
- https://sysdig.com/blog/kubernetes-monitoring-prometheus/
- https://github.com/prometheus-operator/prometheus-operator/blob/master/example/additional-scrape-configs/additional-scrape-configs.yaml
- https://sysdig.com/blog/kubernetes-monitoring-with-prometheus-alertmanager-grafana-pushgateway-part-2/
- https://github.com/grafana/kubernetes-app/issues/35
- https://kubernetes.io/docs/tasks/access-application-cluster/service-access-application-cluster/
- https://sysdig.com/blog/kubernetes-monitoring-prometheus-operator-part3/
- https://sysdig.com/blog/kubernetes-monitoring-prometheus/
- https://www.metricfire.com/blog/deploying-grafana-to-kubernetes/
- https://dev.to/reoring/deploy-prometheus-grafana-to-kubernetes-by-helm-3-1485
- https://dev.to/ko_kamlesh/install-prometheus-grafana-with-helm-3-on-local-machine-vm-1kgj