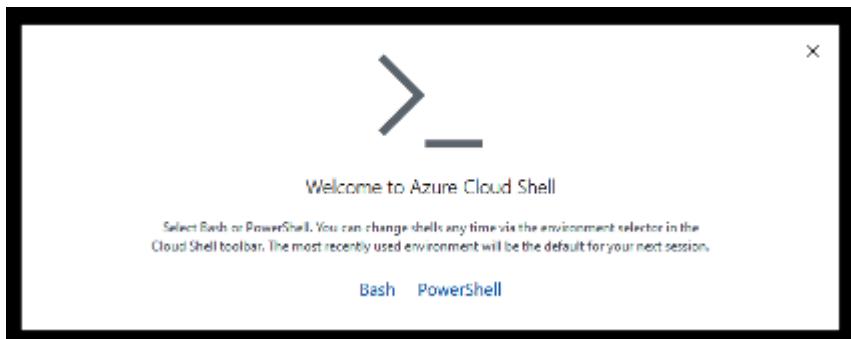


Olá pessoal, hoje venho trazer um simples tutorial de implantação do Elasticsearch no Azure Kubernets AKS, este vai ser um projeto bem simples porem seria legal se você se já tiver algum conhecimento com a stack Elastic.

Vamos lá então.

Entre a CLI do Azure, e vamos usar o Azure Cloud Shell (bash).



```
Bash  |  ?  @  ⌂  ⌂  ⌂  ⌂  ⌂  ⌂
Requesting a Cloud Shell...Succeeded.
Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell

rafael@Azure:~$
```

Em suma, vamos realizar:

- Crie um cluster AKS
- Instale ECK (definições de recursos personalizados + operador)
- Implantar Elastic Stack - Elasticsearch, Kibana

“Antes de prosseguir, lembre-se de que este início rápido faz várias suposições simplificadas que não são recomendadas para uma implantação de produção / séria. Por exemplo, usando credenciais de super usuário para tudo ou desabilitando a validação de TLS, entre outros. Consulte a documentação para obter as melhores práticas.”

Criar cluster AKS

Crie o grupo de recursos do Azure seguido por um cluster AKS.

```
az group create --name eck-quickstart-rg --location westus
```

```
az aks create --resource-group eck-quickstart-rg --name eckAKSCluster --node-count 3 --enable-addons monitoring --generate-ssh-keys --tags "purpose=Elastic Deployment"
```

```
rafael@Azure:~$ az group create --name eck-quickstart-rg --location westus
{
  "id": "/subscriptions/ab2f00b4-5e7f-45e3-ae3b-4790a92b2b1e/resourceGroups/eck-quickstart-rg",
  "location": "westus",
  "managedBy": null,
  "name": "eck-quickstart-rg",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
```

Isso leva alguns minutos. Uma vez feito isso, o próximo ainda será obter as credenciais do cluster para que seu kubectl possa trabalhar com ele.

```
az aks get-credentials --resource-group eck-quickstart-rg --name eckAKScluster
```

```
rafaael@azure:~$ az aks get-credentials --resource-group eck-quickstart-rg --name eckAKSCluster
Merged "eckAKScluster" as current context in /home/rafaael/.kube/config
rafaael@azure:~$
```

Instale ECK

Documentação da Elastic

<https://www.elastic.co/quide/en/cloud-on-k8s/current/k8s-deploy-eck.html>

```
kubectl apply -f https://download.elastic.co/downloads/eck/0.9.0/all-in-one.yaml
```

```
rafael@rafael-OptiPlex-5070: ~ $ kubectl apply -f https://download.elastic.co/downloads/elasticsearch/elasticsearch-1.2.1/all-in-one.yaml
```

Aguarde alguns segundos e você poderá monitorar os registros conforme o operador é aplicado usando o comando abaixo.

```
rafael@Azure:~$ kubectl apply -f https://download.elastic.co/downloads/eck/1.2.1/all-in-one.yaml
customresourcedefinition.apimextensions.k8s.io/apmserver.apps.k8s.elastic.co created
customresourcedefinition.apimextensions.k8s.io/beats.beat.k8s.elastic.co created
customresourcedefinition.apimextensions.k8s.io/elasticsearches.elasticsearch.k8s.elastic.co created
customresourcedefinition.apimextensions.k8s.io/enterprisearches.enterprisesearch.k8s.elastic.co created
customresourcedefinition.apimextensions.k8s.io/kibanas.kibana.k8s.elastic.co created
namespace/elastic-system created
serviceaccount/elastic-operator created
secret/elastic-webhook-server-cert created
clusterrole.rbac.authorization.k8s.io/elastic-operator created
clusterrole.rbac.authorization.k8s.io/elastic-operator-view created
clusterrole.rbac.authorization.k8s.io/elastic-operator-edit created
clusterrolebinding.rbac.authorization.k8s.io/elastic-operator created
rolebinding.rbac.authorization.k8s.io/elastic-operator created
service/elastic-webhook-server created
statefulset.apps/elastic-operator created
validatingwebhookconfiguration.admissionregistration.k8s.io/elastic-webhook.k8s.elastic.co created
rafael@Azure:~$
```

Para ver os logs durante o processo

```
kubectl -n elastic-system logs -f statefulset.apps/elastic-operator
```

Crie o arquivo para o Elasticsearch

Começaremos com Elasticsearch – um cluster de 3 nós com todos os nós assumindo múltiplas responsabilidades (mestre, ingestão, dados). Crie o arquivo `elasticsearch.yaml` com o conteúdo abaixo e salve o arquivo. Sinta-se à vontade para escolher o editor de texto (vi ou nano) de sua escolha.

```
apiVersion: elasticsearch.k8s.elastic.co/v1
kind: Elasticsearch
metadata:
  name: quickstart
  labels:
    component: elasticsearch
spec:
  version: 7.9.3
  http:
    service:
      spec:
        type: LoadBalancer
  nodeSets:
    - name: default
      count: 3
      config:
        node.master: true
        node.data: true
        node.ingest: true
        node.store.allow_mmap: false
```

```
apiVersion: elasticsearch.k8s.elastic.co/v1
kind: Elasticsearch
metadata:
  name: quickstart
  labels:
    component: elasticsearch
spec:
  version: 7.9.3
  http:
    service:
```

```

spec:
  type: LoadBalancer
nodeSets:
- name: default
  count: 3
  config:
    node.master: true
    node.data: true
    node.ingest: true
    node.store.allow_mmap: false

```

O valor *LoadBalancer* para a propriedade *type* atribui um endereço *external-ip* ao serviço *elasticsearch* para que você possa testá-lo em seu navegador.

Isso implantará um cluster Elasticsearch de 3 nós, portanto, pode levar alguns minutos (levou cerca de 3 minutos para mim). Espere até que a coluna *HEALTH* mude de vermelho para verde .

Depois verifique o status do Elasticsearch

```
watch kubectl get elasticsearch
```

```

Every 2.0s: kubectl get elasticsearch

NAME      HEALTH   NODES   VERSION   PHASE   AGE
quickstart  green     1       7.9.3     Ready   113s

```

Agora vamos pegar o IP da nossa instalação

```
kubectl get service quickstart-es-http
```

```

root@Azure:~$ kubectl get service quickstart-es-http
NAME          TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)        AGE
quickstart-es-http   LoadBalancer   10.0.59.242   <none>        9200:32657/TCP   2m26s

```

Bem antes de abrir o navegador e ver o seu Elasticsearch temos de pegar o usuário e senha

```
echo $(kubectl get secret quickstart-es-elastic-user -o=jsonpath='{.data.elastic}' | base64 --decode)
```

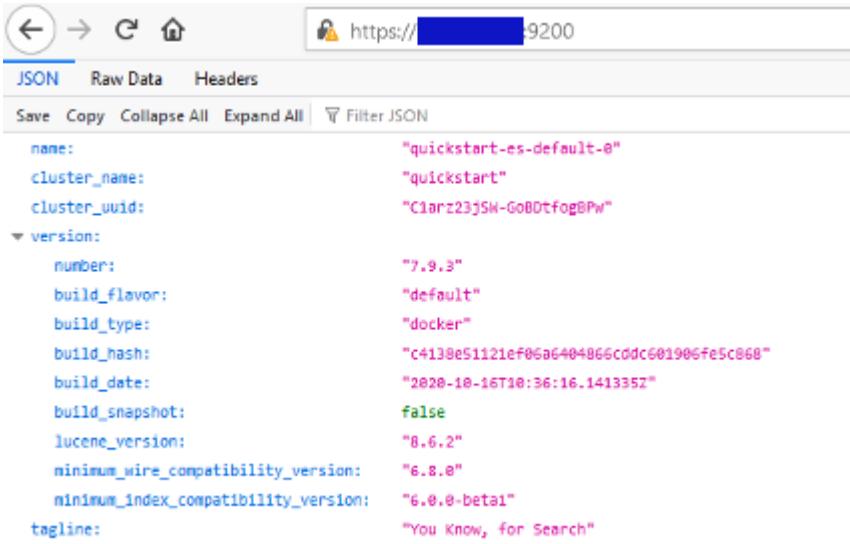
```

root@Azure:~$ echo $(kubectl get secret quickstart-es-elastic-user -o=jsonpath='{.data.elastic}' | base64 --decode)
elastic:elastic
root@Azure:~$ 

```

Sucesso!

OBS. Como o certificado é auto assinado o seu navegador vai apresentar a mensagem se deseja continuar ok...



A screenshot of a web browser displaying a JSON configuration file. The URL in the address bar is `https://[REDACTED]:9200`. The page title is "JSON Raw Data Headers". Below the title are buttons for "Save", "Copy", "Collapse All", "Expand All", and "Filter JSON". The main content is a JSON object representing an Elasticsearch cluster configuration:

```
name: "quickstart-es-default-e"
cluster_name: "quickstart"
cluster_uuid: "Ciarz23jSk-GoBDtfogBPw"
version:
  number: "7.9.3"
  build_flavor: "default"
  build_type: "docker"
  build_hash: "c4130e51121ef06a6404866cddc601906fe5c068"
  build_date: "2020-10-16T10:36:16.141335Z"
  build_snapshot: false
  lucene_version: "8.6.2"
  minimum_wire_compatibility_version: "6.8.0"
  minimum_index_compatibility_version: "6.0.0-beta1"
tagline: "You Know, for Search"
```

Agora vamos criar o Kibana para visualizar as nossas métricas

```
apiVersion: kibana.k8s.elastic.co/v1
kind: Kibana
metadata:
  name: quickstart
  labels:
    component: kibana
spec:
  version: 7.9.3
  count: 1
  elasticsearchRef:
    name: quickstart
  http:
    service:
      spec:
        type: LoadBalancer
    tls:
      selfSignedCertificate:
        disabled: true
```

```
apiVersion: kibana.k8s.elastic.co/v1
kind: Kibana
metadata:
  name: quickstart
  labels:
```

```
component: kibana
spec:
  version: 7.9.3
  count: 1
 .elasticsearchRef:
    name: quickstart
  http:
    service:
      spec:
        type: LoadBalancer
    tls:
      selfSignedCertificate:
        disabled: true
```

Rodando o comando para criar.

```
kubectl apply -f kibana.yaml
```

```
rafael@Azure:~$ kubectl apply -f kibana.yaml
kibana.kibana.k8s.elastic.co/quickstart created
rafael@Azure:~$ []
```

Validando o status

```
watch kubectl get kibana
```

```
Every 2.0s: kubectl get kibana
NAME      HEALTH   NODES   VERSION   AGE
quickstart   red          7.9.3    71s
```

Agora vamos obter o endereço do Kibana (precisamos ver os dados não é)

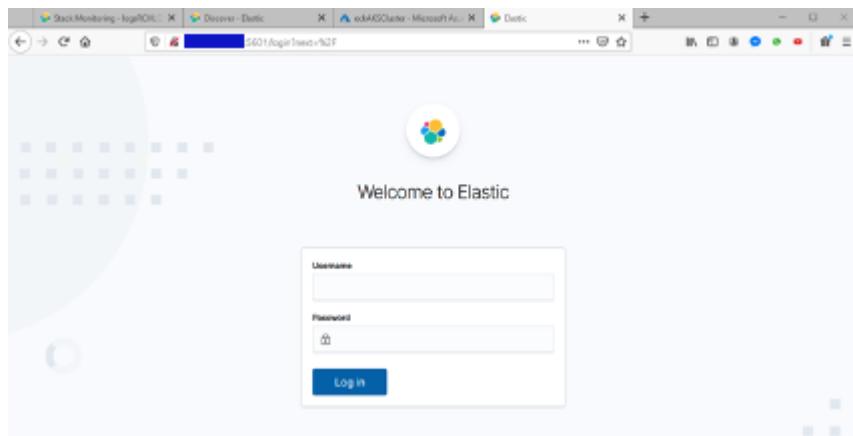
Use o comando:

```
kubectl get service quickstart-kb-http
```

```
rafael@Azure:~$ kubectl get service quickstart-kb-http
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
quickstart-kb-http   LoadBalancer   10.0.104.120   [REDACTED]   5601:30139/TCP   100s
rafael@Azure:~$ []
```

Acessando:

OBS. Lembrem que tirei o https ok.



A screenshot of the Elastic Stack Home dashboard. The top section displays four main modules: 'Observability' (with APM, Logs, Metrics sections), 'Security' (with SIEM + Endpoint Security section), 'Visualize and Explore Data' (with APM and App Search sections), and 'Manage and Administer the Elastic Stack' (with Console and Rollups sections). Below these modules are three buttons: 'Add sample data', 'Upload data from log file', and 'Use Elasticsearch data'.

Bem agora vamos encaminhar alguns logs

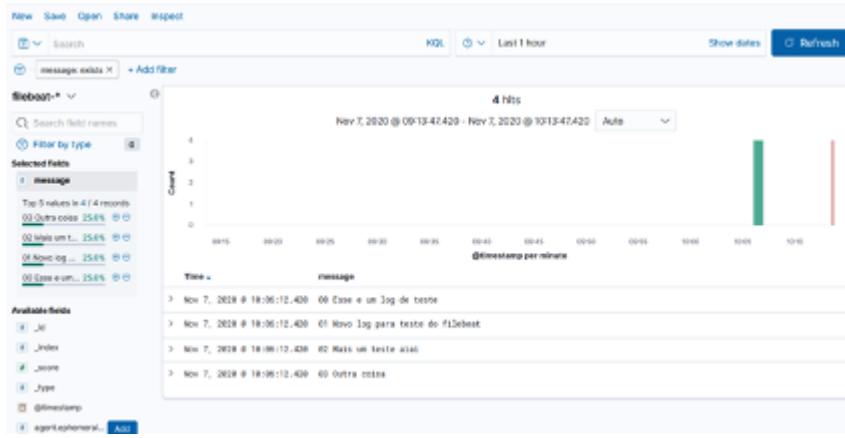
Aqui vou encaminhar algumas informações do filebeat (Já existe um artigo ensinando como usar [AQUI!](#)!).

Uma nota você precisa desabilitar a verificação de ssl ok

Adicione essa linha na configuração para o Elasticsearch

```
ssl.verification_mode: none
```

Segue exemplo do meu log “bem simples”



Então é isso pessoal espero que possa ter ajudado um pouco com esse simples tutorial e qualquer dúvida não deixe de comentar será um prazer ajudar.

Sem mais fiquem com Deus e com objetivos claros. ☺