Kubernetes Commands

Get All Secrets

```
kubectl get secrets -n elasticsearch
```

Get the content of specific secret

```
kubectl get secret elasticsearch-master-certs -n elasticsearc
h -o jsonpath='{.data}' | jq -r 'keys[]'
```

Check the mounting is correct

```
kubectl exec -it -n elasticsearch logstash-logstash-0 -- ls /
usr/share/logstash/config/certs
```

Check if got Logstash certificate issue

```
 kubectl exec -it -n elasticsearch logstash-logstash-0 -- cur
l -v -k -u elastic:FvPtwA0qqg4H1S8R https://elasticsearch-mas
ter:9200

 Correct output
* successfully set certificate verify locations:
*   CAfile: /etc/ssl/certs/ca-certificates.crtd;e


v
```

Apply changes of a file

```
kubectl apply -f file.yaml -n elasticsearch
```

Delete pod

```
kubectl delete pod -l app=logstash-logstash -n elasticsearch
```

Restart Statefulset

```
kubectl rollout restart statefulset -n elasticsearch logstash
-logstash
```

Check Logs

```
kubectl logs logstash-logstash-0 -n elasticsearch
```

Pull the latest changes

```
kubectl get service logstash-logstash-headless -n elasticsear
ch -o yaml > latest-service.yaml
```

Uninstall logstash

```
helm uninstall logstash  -n elasticsearch
helm dep build logstash-parent/ -n elasticsearch
helm install logstash logstash-parent/ -n elasticsearch
kubectl get pod -n elasticsearch -w
kubectl logs logstash-logstash-0 -n elasticsearch
```

To got All Services

```
kubectl get services -n elasticsearch
```

To get Password

```
kubectl get secret elasticsearch-master-credentials -n elasti
csearch -o go-template='{{.data.password | base64decode }}'
```

Check Kubernetes Nodes status

```
kubectl get nodes
```

Check the memory usage

```
kubectl describe node master-node
```

Check Elasticsearch Pods

```
kubectl get pods --namespace=elasticsearch -w
```

To Check Description of PVC

```
Fkubectl describe pvc elasticsearch-master-elasticsearch-mast
er-0
```

Get all PVCs

```
kubectl get pvc --namespace=elasticsearch
```

Check Readiness Probe

```
kubectl exec elasticsearch-master-0 -n elasticsearch -- curl
-k -u elastic:$(kubectl get secret elasticsearch-master-crede
ntials -n elasticsearch -o jsonpath='{.data.password}' |base6
4 --decode) https://localhost:9200/_cluster/health?pretty
```

Verify Resource Applied

```
kubectl get statefulset elasticsearch-data -n elasticsearch -
o yaml | grep -A 5 resources:
```

Delete PVc for Pod

```
kubectl delete pvc elasticsearch-master-elasticsearch-master-
```

```
0 --namespace=default
```

Check PV assigned PVC

```
kubectl get pods -n elasticsearch -o custom-columns=NAME:.met
adata.name,PVC:.spec.volumes[*].persistentVolumeClaim.claimNa
me

Or

kubectl get pvc -n elasticsearch
```

Upgrade statefulset after making changes to values.yaml file

```
helm upgrade elasticsearch elastic/elasticsearch -f ./values.
yaml --namespace=elasticsearch
```

To check a specific pod yaml file

```
kubectl get pod elasticsearch-master-0 --namespace=elasticsea
rch -o yaml
```

To create new directory

```
mkdir -p ~/k8s-manifests/storage
```

To Uninstall Kibana

```
helm uninstall kibana -n elasticsearch --no-hooks
kubectl delete deployment,statefulset,service,ingress,configm
ap,secret,serviceaccount,role,rolebinding,pvc,job,cronjob -l
app=kibana -n elasticsearch
kubectl delete pod -l app=kibana -n elasticsearch --force --g
race-period=0
kubectl get pv | grep kibana | awk '{print $1}' | xargs -I {}
kubectl delete pv {}
```

```
kubectl delete secret -n elasticsearch -l name=kibana
kubectl get all,configmap,secret,pvc,serviceaccount,role,role
binding -l app=kibana -n elasticsearch
```

To install helm values file

```
helm show values elastic/elasticsearch > values.yaml
helm show values elastic/kibana > kibana-values.yaml
helm show values elastic/logstash > logstash-values.yaml
```

Got pod describtion

```
kubectl describe pod elasticsearch-master-0 --namespace=elast
icsearch
```

Check Previous logs

```
kubectl logs elasticsearch-master-0 --namespace=elasticsearch
--previous
```

Scale up or Down statefulset

```
kubectl scale statefulset elasticsearch-master --namespace=de
fault --replicas=1
```

To verify the CPU and Memory of each pod

```
kubectl top pod -n elasticsearch
```

Overall node health

```
kubectl top node
```

How to Query Elasticsearch From Python Script

1- Find elasticsearch-master cluster ip address via

```
kubectl get svc -n elasticsearch
```

Add clusterip elasticsearch-master to /etc/hosts

```
10.101.131.136 elasticsearch-master
```

1. **Create the Directory (if it doesn't exist)**:

   ```
   mkdir -p ~/certs
   ```

2. **Copy the CA Certificate**: (How to copy Certificate to physical folder)

   ```
   kubectl cp elasticsearch/logstash-logstash-0:/usr/share/lo
   gstash/config/certs/ca.crt ~/certs/ca.crt
   ```

3. Run the script

```
from elasticsearch import Elasticsearch
from datetime import datetime

def get_if_device_status(ip_address):
    # Connect to Elasticsearch using the hostname
    es = Elasticsearch(
        ['https://elasticsearch-master:9200'],  # Use the hos
tname of the Elasticsearch service
        basic_auth=('elastic', 'JQvEBqNz5zV4kVOM'),
        verify_certs=True,
        ca_certs="/home/ubtadmin/certs/ca.crt"  # Update this
path to the local path of your CA certificate
    )

    # Define the search query with sorting by @timestamp in d
escending order and script_fields
```

```
    query = {
        "_source": ["ifDeviceStatus", "@timestamp"],  # Ensur
e _source fields are included
        "query": {
            "match": {
                "host.ip": ip_address
            }
        },
        "sort": [
            {
                "@timestamp": {
                    "order": "desc"
                }
            }
        ],
        "script_fields": {
            "local_timestamp": {
                "script": {
                    "lang": "painless",
                    "source": "doc['@timestamp'].value.plusHo
urs(8)"
                }
            }
        }
    }

    # Execute the search query
    response = es.search(index="snmp-data-*", body=query)

    # Extract and print the ifDeviceStatus and local_timestam
p fields from the results
    for hit in response['hits']['hits']:
        if '_source' in hit and 'ifDeviceStatus' in hit['_sou
rce']:
            local_timestamp = hit['fields']['local_timestam
p'][0] if 'fields' in hit and 'local_timestamp' in hit['field
```

```
s'] else 'No timestamp'
            if local_timestamp != 'No timestamp':
                # Parse the local timestamp
                local_timestamp = datetime.strptime(local_tim
estamp, "%Y-%m-%dT%H:%M:%S.%fZ")
                formatted_timestamp = local_timestamp.strftim
e("%Y-%m-%d %H:%M:%S")
            else:
                formatted_timestamp = 'No timestamp'
            print(f"IP: {ip_address}, ifDeviceStatus: {hit['_
source']['ifDeviceStatus']}, Local Timestamp: {formatted_time
stamp}")
        else:
            print(f"IP: {ip_address}, ifDeviceStatus: Not fou
nd, Local Timestamp: No timestamp")

# Example usage
get_if_device_status("172.22.24.168")
```

To restart kibana

```
kubectl rollout restart deployment -n elasticsearch kibana
```

To start Kubernetes Dashboard forward the service via

```
kubectl port-forward service/kubernetes-dashboard-kong-proxy -n
```

Get the token with

```
kubectl -n kubernetes-dashboard create token admin-user
```

The these commands exists on a file of eckconfig → dsashboard_token.sh and kube_dashboard.sh