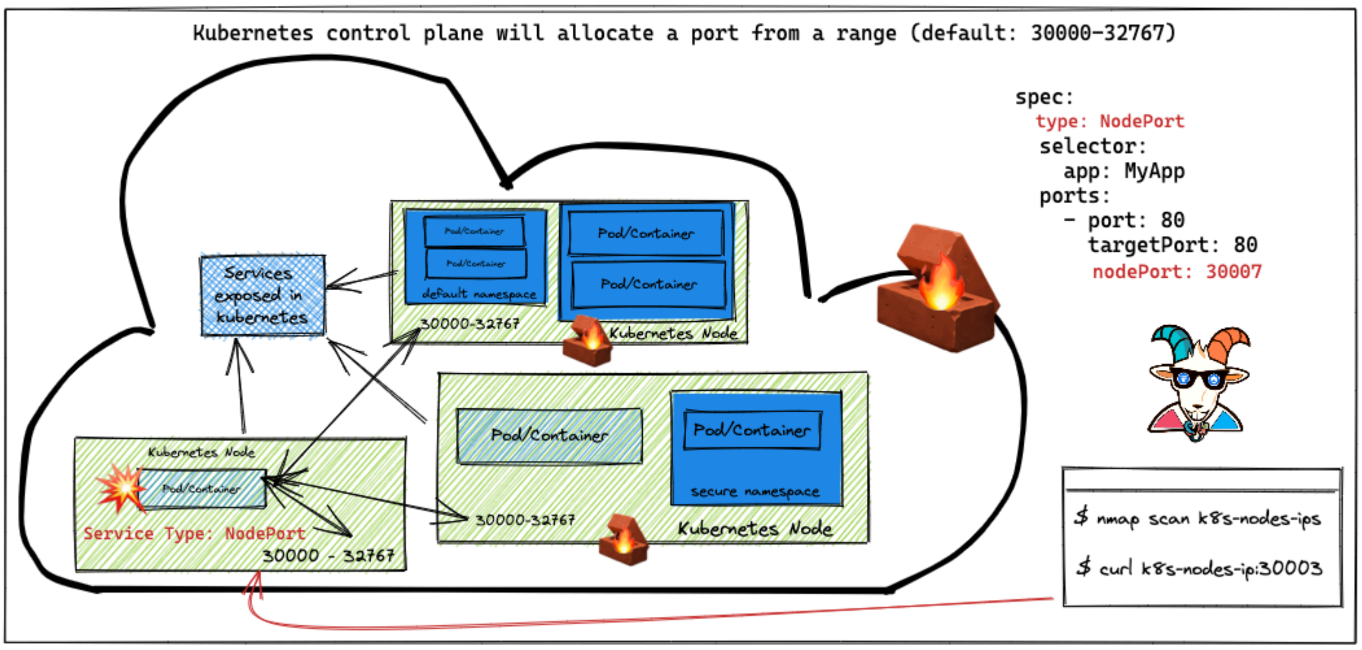
⎈ NodePort exposed services

🙌 Overview

In this scenario, we see another misconfiguration that may give attackers access to internal services and non-exposed services. This is one of the simple misconfigurations made when creating the Kubernetes services and also the cluster setup and configurations.


Kubernetes control plane will allocate a port from a range (default: 30000-32767)

```
spec:
  type: NodePort
  selector:
    app: MyApp
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30007
```

```
$ nmap scan k8s-nodes-ips
$ curl k8s-nodes-ip:30003
```

By the end of the scenario, we will understand and learn the following

1. How Kubernetes `NodePort` works, configuration, and the ranges
2. Performing the portscan and interacting with Kubernetes nodes

⚡ The story

If any of the users exposed any service within the Kubernetes cluster with `NodePort`, this means that the nodes where the Kubernetes clusters are running don't have any firewall/network security enabled. We need to see some unauthenticated and unauthorized services.

INFO

To get started with the scenario, run the following command and look for Kubernetes nodes external IP addresses

```
kubectl get nodes -o wide
```

TIP

When Kubernetes creates a `NodePort` service, it allocates a port from a range specified in the flags that are defined in your Kubernetes cluster configuration. (By default, these are ports ranging from 30000-32767.)

🎯 Goal

TIP

If you can access the metadata-db service using the public IP address of the node with the NodePort, then you can complete the scenario.

Hints & Spoilers

▸ ✨ Still looking at the Nodes Public IPs?

🎉 Solution & Walkthrough

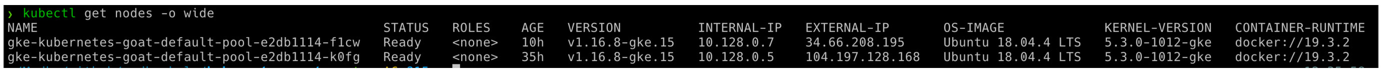
🎲 Method 1

CAUTION

This vulnerability/attack varies depending on how the Kubernetes cluster has been configured. If your Kubernetes cluster is configured with firewalling and blocking the nodes exposed ports, then this scenario might not work for you but the concepts are the same 😊

- Get the list of Kubernetes nodes external IP addresses information by running the following command

```
kubectl get nodes -o wide
```



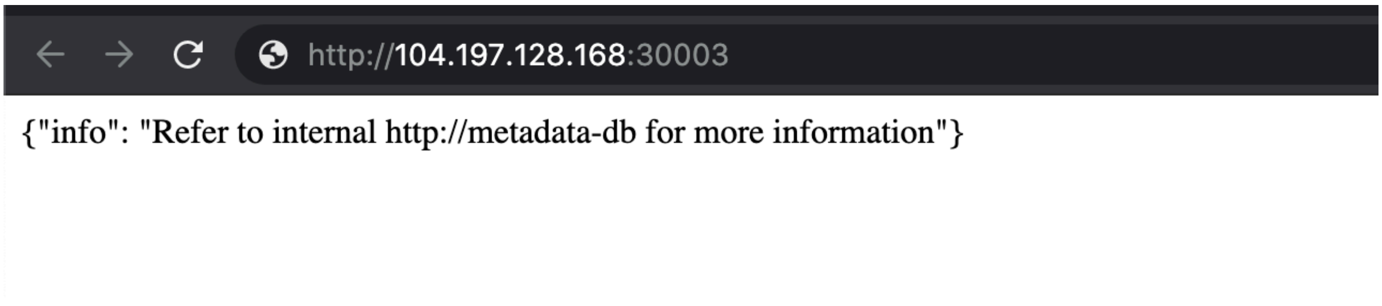
```
› kubectl get nodes -o wide
NAME                                              STATUS   ROLES    AGE   VERSION            INTERNAL-IP   EXTERNAL-IP       OS-IMAGE             KERNEL-VERSION     CONTAINER-RUNTIME
gke-kubernetes-goat-default-pool-e2db1114-f1cw    Ready    <none>   10h   v1.16.8-gke.15     10.128.0.7    34.66.208.195     Ubuntu 18.04.4 LTS   5.3.0-1012-gke     docker://19.3.2
gke-kubernetes-goat-default-pool-e2db1114-k0fg    Ready    <none>   35h   v1.16.8-gke.15     10.128.0.5    104.197.128.168   Ubuntu 18.04.4 LTS   5.3.0-1012-gke     docker://19.3.2
```

TIP

Now, let's find out the open ports. In this case, you can use your traditional security scanning utilities like `nmap`

- Once we identified that there is a NodePort exposed, we can just verify by connecting and accessing it

```
nc -zv EXTERNAL-IP-ADDRESS 30003
```



http://104.197.128.168:30003

```
{"info": "Refer to internal http://metadata-db for more information"}
```

- Hooray 🥳 , now we can see that we can access the internal services which are not exposed to the world can be accessed and bypassed due to the `NodePort` configuration

References

- Kubernetes Services - NodePort

Edit this page