Exposed Kubernetes Clusters

June 27, 2022

Organizations At Risk Of Data Breaches Via Misconfigured Kubernetes

During our routine threat-hunting exercise, Cyble Research Labs observed over 900,000 Kubernetes exposures across the internet . This does not necessarily imply that all exposed instances are vulnerable to attacks or will lead to the loss of sensitive data. Rather, it emphasizes the existence of seemingly simple misconfiguration practices that might make companies lucrative targets for TAs in the future.

Kubernetes, often known as K8s, is an open-source system for automating containerized application deployment, scaling, and administration. K8s incorporates virtual and real machines to create a uniform API interface. Developers can use the Kubernetes API to launch, scale, and manage containerized applications. Kubernetes assists with the management of containers that execute applications and ensures that there is no downtime in a production environment.

Previously, the exposure of Kubernetes consoles over the internet has been the primary reason for multiple incidents, including organizations like Tesla, where hackers had infiltrated Tesla's Kubernetes console, which was not password protected. Within one Kubernetes pod, access credentials were exposed to Tesla's AWS environment, which contained an Amazon S3 (Amazon Simple Storage Service) bucket containing sensitive data such as telemetry.

Online scanners have made it easy for security researchers to find the exposure of assets. Regardless, at the same time, malicious hackers can also investigate the exposed Kubernetes instance for a particular organization, increasing the risk of attack. The below queries can be used to find exposure for Kubernetes instances exposed over the internet:

- KubernetesDashboard
- Kubernetes-master
- Kubernetes
- Kube
- K8
- Favicon:2130463260, -1203021870

While investigating the exposure of Kubernetes instance, it was noticed that the United States has the highest exposure count, followed by China and Germany, as shown below. Below, we have showcased some stats based on the query "*product:Kubernetes*."

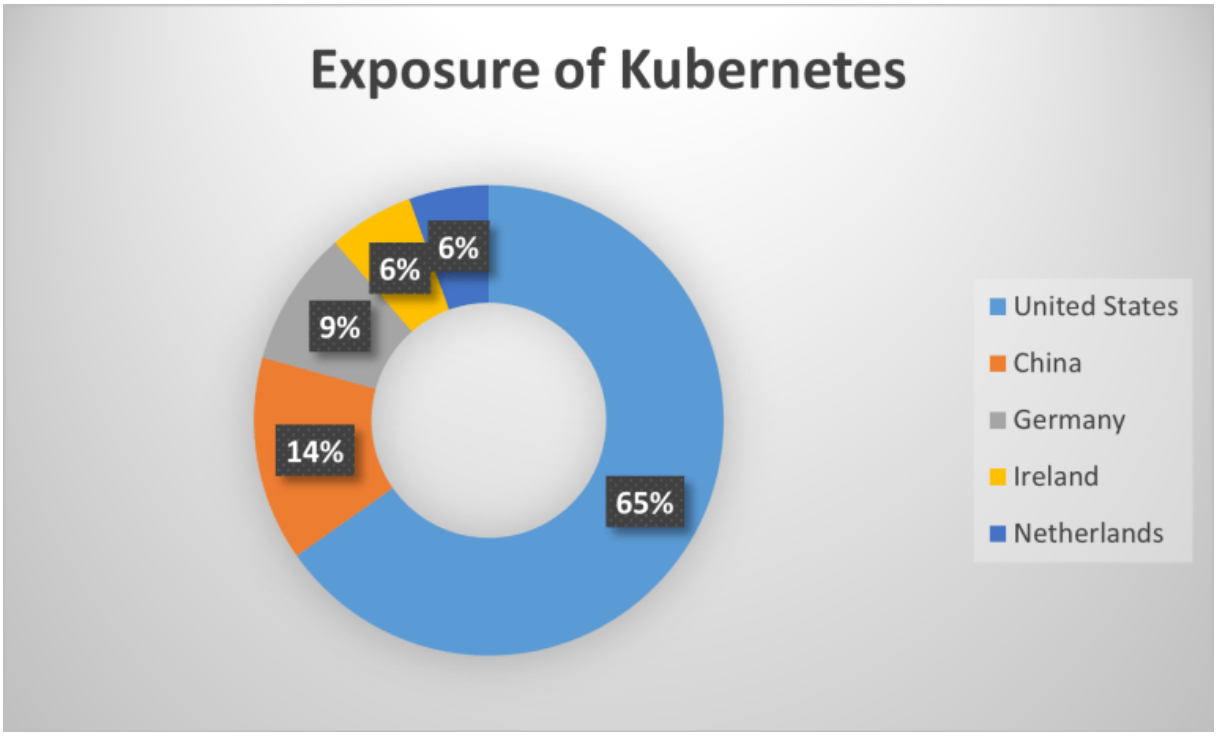

Figure 1 – Kubernetes Exposure by Country

The top 3 exposed ports discovered during the investigation are shown below.

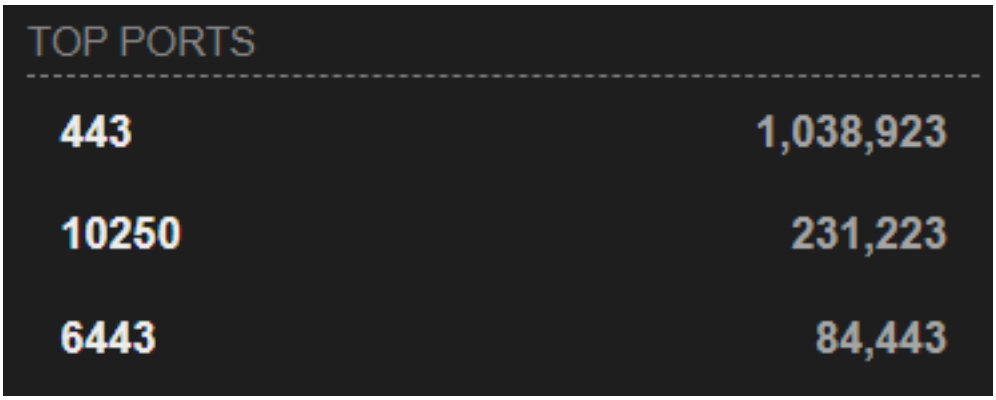

Figure 2 – Exposure of top 3 ports

Kubernetes Architecture

Master Node

Through an API, the Kubernetes Master (Master Node) accepts input through a CLI (Command-Line Interface) or UI (User Interface). These are the commands that users provide to Kubernetes.

API Server

The API Server is the control plane's front-end and the sole component with which users interface directly. Internal system components and external user components both communicate using the same API.

Scheduler

A Scheduler monitors new API Server requests and allocates them to healthy nodes. It rates the nodes' quality and delivers pods to the best-suited node.

Controller Manager

The Controller's job is to get the desired status from the API Server. It examines the present state of the nodes it is tasked with controlling, assesses whether there are any disparities, and resolves them.

etcd

The Key-Value Store, also called **etcd**, is a database that Kubernetes uses to back up all cluster data. It stores the entire configuration and state of the cluster. The Master node queries **etcd** to retrieve parameters for the state of the nodes, pods, and containers.

Worker Node

Worker nodes monitor the API Server for new task assignments, perform them, and then return the results to the Kubernetes Master node.

Kubelet

Every node in the cluster executes the Kubelet. It serves as the primary Kubernetes agent. It monitors the API Server for tasks, executes them, and reports back to the Master Node. It also monitors pods and notifies the control panel if one is not completely working. The Master Node can then determine how to assign tasks and resources based on this knowledge in order to achieve the desired state.

Kube-Proxy

The Kube-proxy ensures that each node receives an IP address and that local iptables and rules are in place to manage routing and traffic load-balancing.

Pods

A pod is the smallest scheduling element in Kubernetes. A container cannot be a member of a cluster without it. If you need to grow your app, the only way to do it is to add or remove pods.

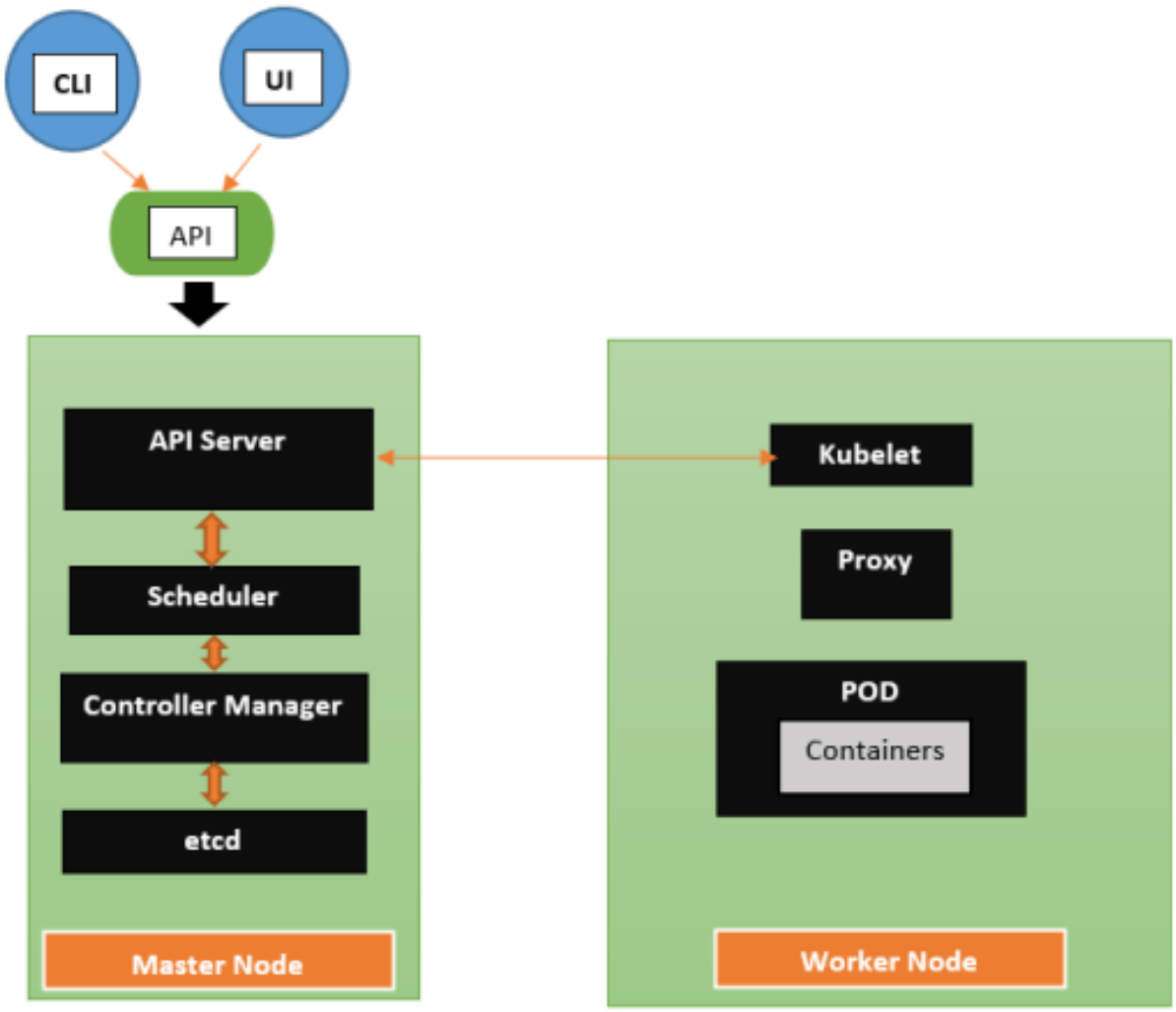

Figure – 3 Kubernetes Architecture

Status Codes Statistics

- During our investigation using the query “Kubernetes,” it was observed that most of the exposuress have the status code 403.

This signifies that the Kubelet API accepted the unauthenticated request but determined that we lacked the necessary rights (authorization) to visit that endpoint, as shown in the figure below.


```
{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {},
  "status": "Failure",
  "message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",
  "reason": "Forbidden",
  "details": {},
  "code": 403
}
```


Figure 4 – Status 403 response

- As shown in the figure below, a few of the exposures returned status code 401, as shown in Figure 5. This information implies that a Kubernetes cluster is functioning in the environment. This may lead to the attacker attempting various K8s exploits and vulnerabilities to enter the environment.

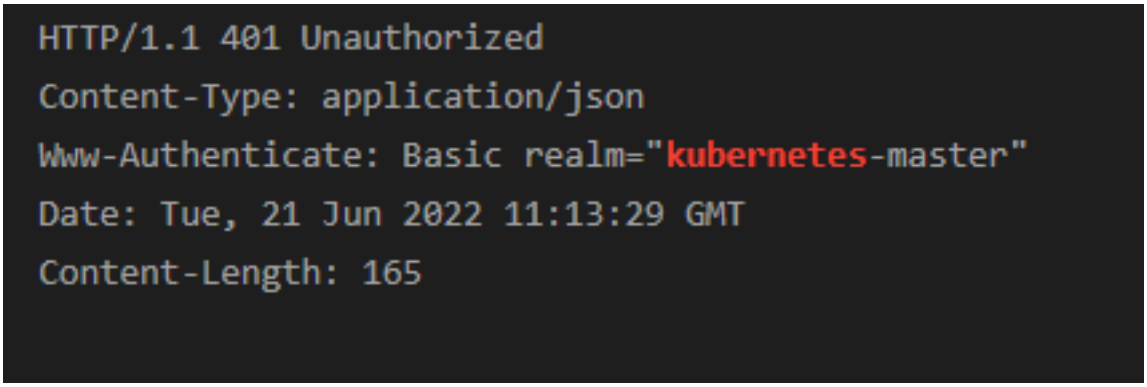

Figure 5 – Status code 401

- Some exposed instances returned status code 200, indicating that certain Kubelet-running nodes responded with information about which pods were running on that node.

Attackers using online scanners can investigate the Kubernetes exposure of an organization and might be able to find Kubernetes Dashboard, which is not password-protected, as shown in the figure below.

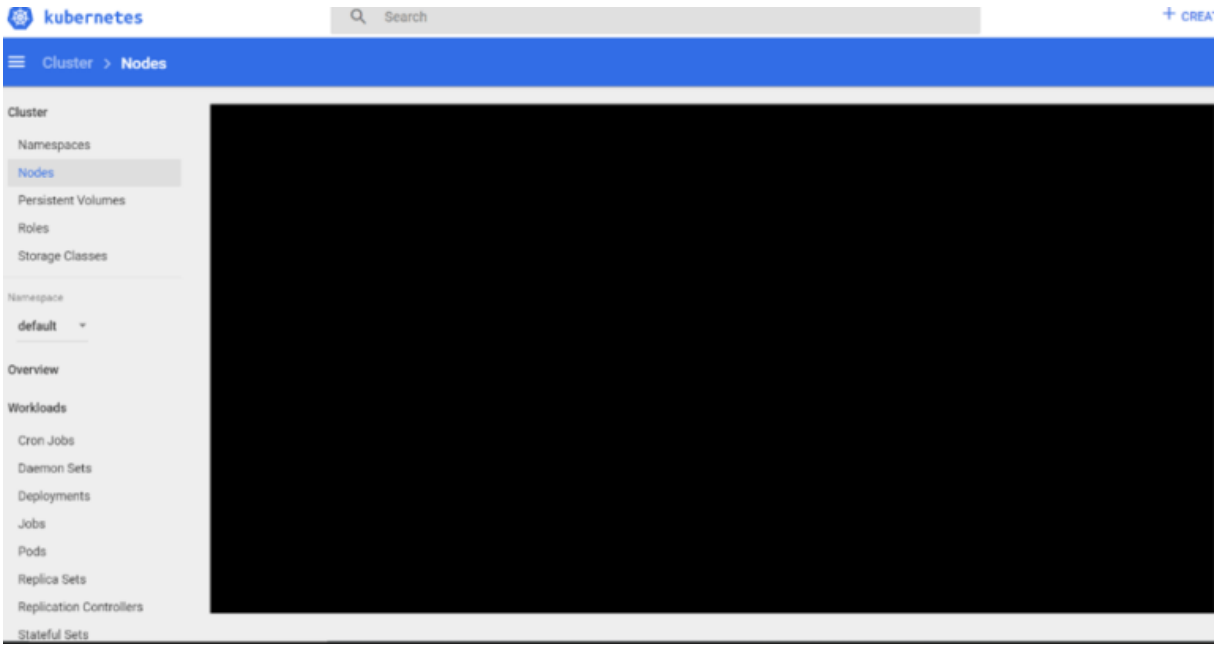

Figure 6 – Kubernetes Exposed Dashboard example

The table below provides the count of exposures based on their status codes.

Status Code	Count
403	975,371
401	4,954
200	799

Conclusion

With the popularity and use of Kubernetes, users must be careful about their cybersecurity posture. For enterprises, default configurations can pose substantial security threats.

Misconfigurations like utilizing default container names, not having the Kubernetes Dashboard protected by a secure password and leaving default service ports open to the public can place businesses at risk of data leakage.

Recommendations

1. Keep Kubernetes updated to the latest version.
2. Make certain to remove debugging tools from production containers.
3. Control individuals with access to the Kubernetes API and what permissions they have with Role-Based Access Control (RBAC).
4. Limit exposure of critical assets and ports. Ensure that your network restricts access to kubelet-related ports, such as 10255 and 10250. Consider restricting access to the Kubernetes API server to trustworthy networks only.

5. To minimize the possible impact of a breach, important workloads should be executed on a separate group of workstations. This method decreases the possibility of a sensitive application being accessed via a less-secure application that shares a container runtime or host.
6. Follow Kubernetes configuration best practices.
7. Create and define cluster network policies and cluster-wide Pod Security Policy.
8. CIS Kubernetes Benchmark should be the first step before going through any specific Kubernetes security issues or security improvements.
9. Ensure that audit logs are enabled and are being monitored for unusual API requests, particularly any permission failures.
10. Minimize administrative access to Kubernetes nodes.