Kubernetes Node Attacks

Attacking the Cluster from the Nodes

Handling the Compromised Node Scenario

Node Attacks

An attacker can gain access to a node through at least three different measures:

- Break out of a container via an exploit
- Phish an engineer that has login access to the node
- Use an authorization weakness in the Kubernetes cluster
- Compromise a container that has too much privilege

The last two of these look very much the same.

Overprivileged Containers

If a bad actor can find or create a container that has too much privilege, they can compromise the node or the cluster.

Here are three examples:

- “privileged” containers have no capability limits and mount the entire /dev tree.
- “hostNetwork” containers use the node’s network namespace.
- Containers that mount the node’s filesystem … have access to the node’s filesystem.

Privileged Containers

Privileged containers are particularly powerful.

Here are the salient points from an attacker's perspective.

- A privileged container mounts the entire /dev tree.
- It can insert a module into the running kernel.
- It has access to every root capability, rather than the reduced set afforded a container.

We'll use two of these in our node attacks exercise.

Privileged Container Device Directory

Standard Container's /dev Directory

```
root@ae8d2c3a5ed4:/# ls /dev
core  fd  full  mqueue  null  ptmx  pts  random  shm  stderr  stdin  stdout  tty  urandom  zero
```

Privileged Container's /dev Directory

```
root@b3a43968ece6:/# ls /dev
autofs           fuse          mqueue  rfkill    tty1   tty19  tty28  tty37  tty46  tty55  tty7     vcs    vcsa3  vcsu6        vport1p1
btrfs-control    hidraw0       net     rtc0      tty10  tty2   tty29  tty38  tty47  tty56  tty8     vcs1   vcsa4  vda          vport1p2
bus              hpet          null    shm       tty11  tty20  tty3   tty39  tty48  tty57  tty9     vcs2   vcsa5  vda1         watchdog
core             hwrng         nvram   snapshot  tty12  tty21  tty30  tty4   tty49  tty58  ttyS0    vcs3   vcsa6  vda14        watchdog0
cpu_dma_latency  input         port    snd       tty13  tty22  tty31  tty40  tty5   tty59  ttyS1    vcs4   vcsu   vda15        zero
cuse             kmsg          ppp     stderr    tty14  tty23  tty32  tty41  tty50  tty6   ttyS2    vcs5   vcsu1  vfio
dri              kvm           psaux   stdin     tty15  tty24  tty33  tty42  tty51  tty60  ttyS3    vcs6   vcsu2  vga_arbiter
fb0              loop-control  ptmx    stdout    tty16  tty25  tty34  tty43  tty52  tty61  uhid     vcsa   vcsu3  vhci
fd               mapper        pts     tty       tty17  tty26  tty35  tty44  tty53  tty62  uinput   vcsa1  vcsu4  vhost-net
full             mem           random  tty0      tty18  tty27  tty36  tty45  tty54  tty63  urandom  vcsa2  vcsu5  vhost-vsock
```

hostNetwork Pods

hostNetwork pods allow all containers within them to use the host's network namespace, rather than a separate one, the way normal containers do.

This allows the container to impersonate the node, from a network perspective.

If time permits, we'll demonstrate how this can defeat the Kube2IAM security control.

Exercise: Kubernetes Node Attacks

Please:

Open the Firefox browser on the class machine to:
http://localhost:10000/exercises/kubernetes-node-attacks