black hat EUROPE 2023

DECEMBER 4-7

EXCEL LONDON / UK

sysdig narf

black hat EUROPE 2023

Orchestrate This! Kubernetes Rootkit

A collaboration between Sysdig, Narf Industries, and Croatia's finest CTFer

Nicholas Lang, Andrew Hughes and Hrvoje Mišetić

Overview

- Background Info
  - Rootkits
  - Kubernetes
- Our rootkit
  - Kubekit API
  - Docker mode
  - K8s mode
- Detection, DevOps Best Practices
- Q&A + Outro

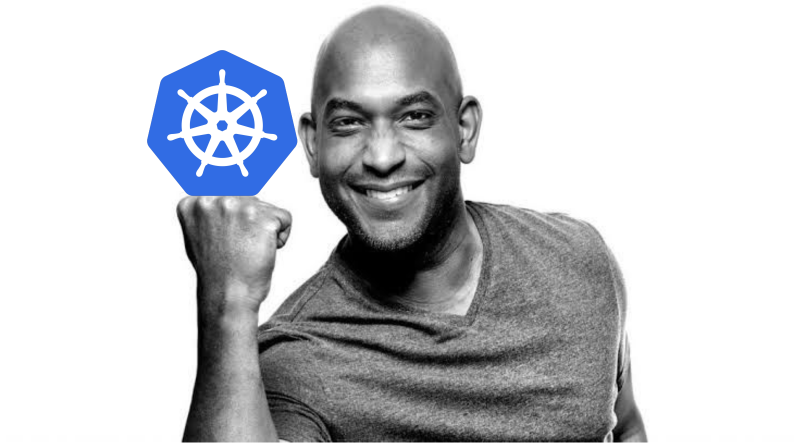
Rootkit in 30 seconds

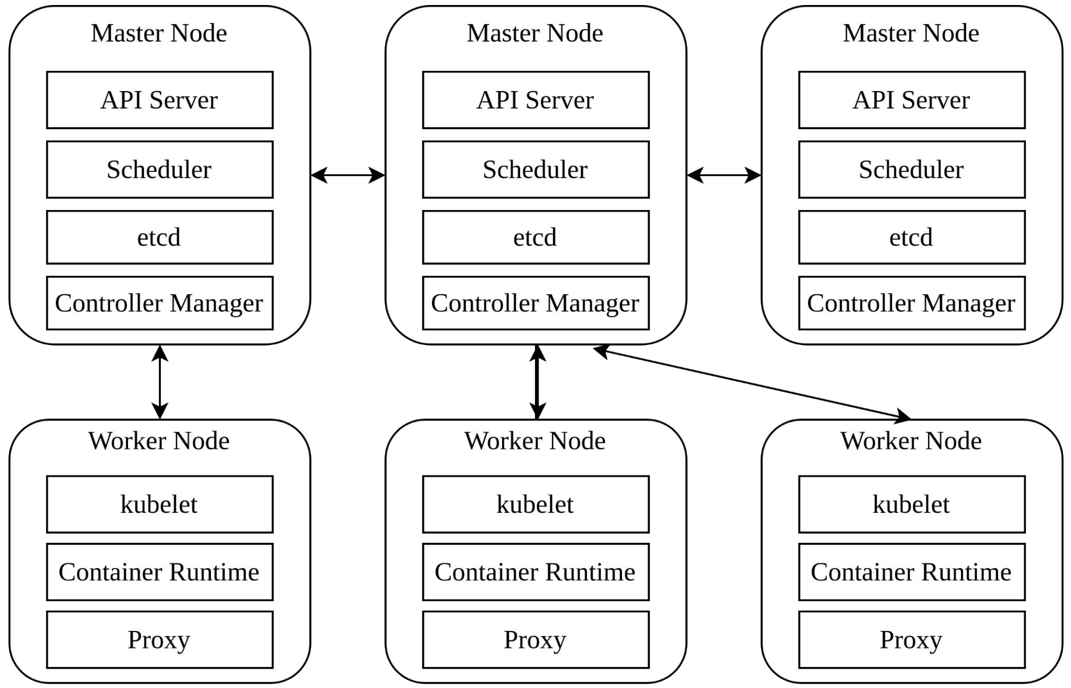
- What can a rootkit do?
  - Stealth
  - Persistence
  - C2 in some cases
  - Basically anything!
- Requires privilege, hence 'root' in rootkit
  - Can be purely in the kernel or have userspace components, common thread is root privilege
- Kernel modules, LD_PRELOADed libraries, custom implementations

Rootkit Name	Mode	Wat do?
Diamorphine	Kernel	hide processes
libprocesshider	userland (LD_PRELOAD)	hide process(es)

Our rootkit guys. Jamie Butler (FU Rootkit - Windows), top. Michael Clark (???, Sebek - Linux), bottom.

Kubernetes 101





- Orchestrates containers using a (container) runtime (containerd, CRI-O, Docker).
- Give it some `yaml` that describes what services you would like and Kubernetes will do the heavy lifting of deploying, load-balancing, and health checking across your cluster (or cloud).
- K8s namespaces vs kernel namespaces
- Control plane vs data plane
  - Control plane: starts, manages, maintains consistency of running pods
  - Data plane: where applications run inside of pods (containers)

Kubekit

Our rootkit!

Goal: Enable purple/red team to be sneakier, forcing blue teams to catch up (have you heard this before?)

Given portable container runtimes to deploy and run payloads: Kubekit is a rootkit that hides malicious containers from their runtime platforms (in this case, Kubectl and docker). **Two modes** for **two** Kubernetes **planes**.

Platform	Kubernetes Version	Docker version
Linux 5.4+	1.28.{3,4}	24.0.5

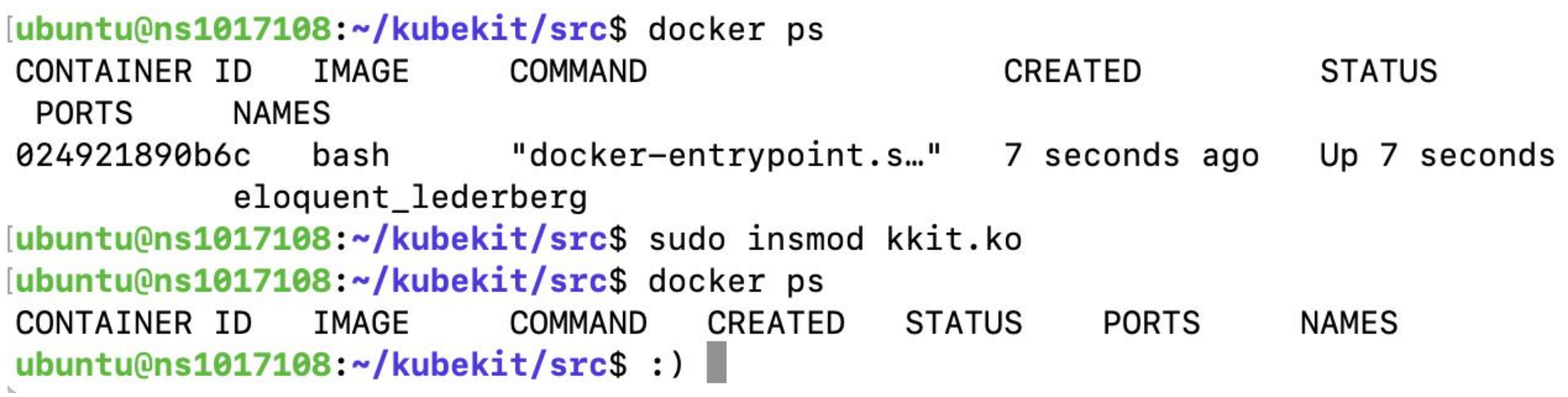
The Kubekit API

- BIG Shoutout to Hrvoje
- Set breakpoints in arbitrary files across namespaces to edit memory.
    - Even has symbol resolution if you don’t know your offsets!
- Hide many fun things unless an environmental variable is set
    - Files
    - Kernel modules - including kubekit
    - Processes
- Cleanly remove any of these features

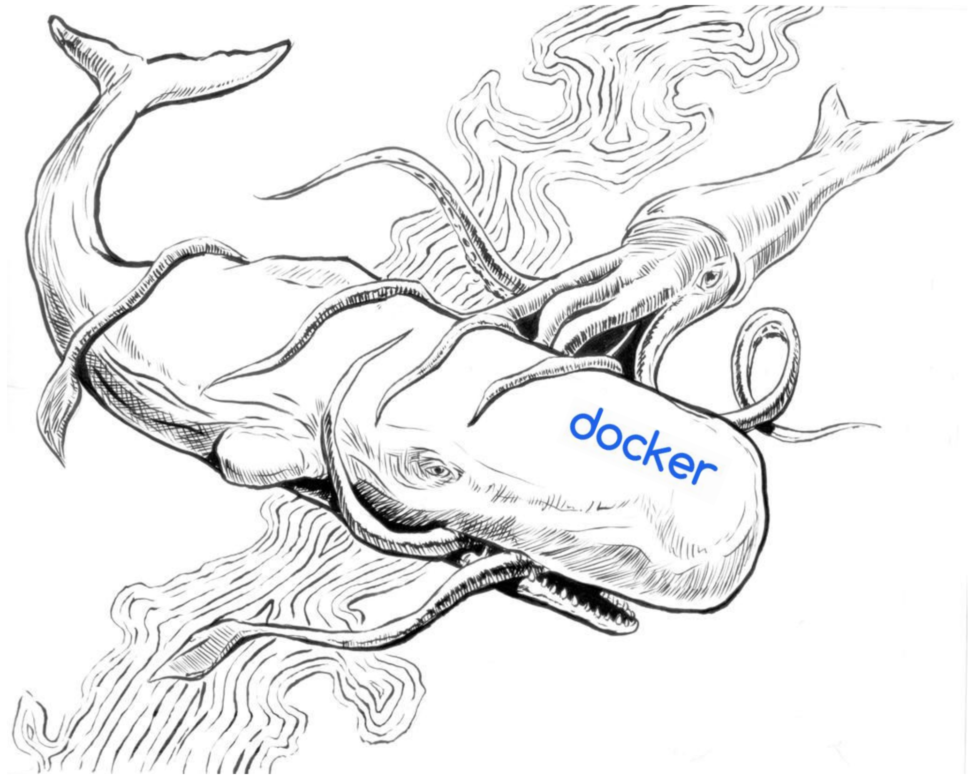
```
func_off = kubekit_find_offset(
    "/usr/bin/dockerd",
    "github.com/docker/docker/daemon.includeContainerInList",
    false
);
if (func_off > 0) {
    ret = kubekit_add_bp(
        "/usr/bin/dockerd",
        func_off,
        FUNC_START,
        docker_include_handler,
        KKFL_PROC
    );
    if (ret == 0) {
        kubekit_log("Added the breakpoint\n");
    }
}
```

Kubekit - Docker

- Why? Docker is very common - 1k exposed (non-TLS) endpoints on shodan
- Why? Docker is very easy to use, package up your bad-boy container and let 'er rip
- Wat do? Hide containers from `docker` management layer
- How? Super awesome breakpointing!!



```
ubuntu@ns1017108:~/kubekit/src$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS
  PORTS     NAMES
024921890b6c   bash      "docker-entrypoint.s…"   7 seconds ago   Up 7 seconds
          eloquent_lederberg
ubuntu@ns1017108:~/kubekit/src$ sudo insmod kkit.ko
ubuntu@ns1017108:~/kubekit/src$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ns1017108:~/kubekit/src$ :)
```



Kubekit - Kubernetes

- Targeting control plane nodes, editing Kubernetes API responses to hide payload pods deployed on nodes
- Hides a userspace components to parse and modify JSON response live
- Hides the target pod from `kubectl get pods` as well as all internal services which use the `/pods` API schema
- This is where the “Orchestrate This!” really comes into play
- **Kubernetes is orchestrating, health-checking, and load-balancing our evil container without sysadmins being able to see it**

Demo

Detecting Kubekit

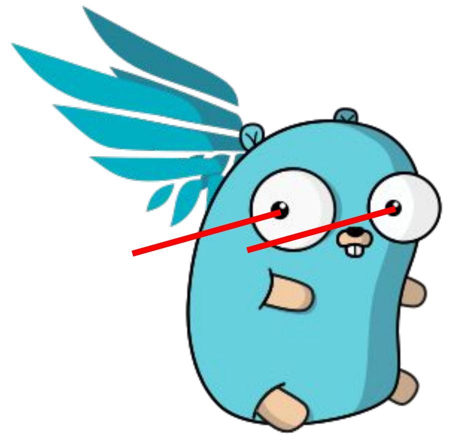
Best practices for k8s sysadmins

- nothing talking to k8s from outside your company network - reduce exposure!
- turn on all of the “hardening” features - even if they’re painful
- SELinux, Access Controls, etc

How you would detect something like this

- kernel modules & logging - **yes you need a rootkit to beat a rootkit**
- eBPF & logging

Example freebie setup **you can do in one afternoon**: `falco` + `falcosidekick` + custom rule to detect the kubekit kmod loading, `falcosidekick` then takes a response action



Why it matters

- Kubernetes is not (sadly) leaving any time soon, even if Hightower did ;)
- Cloud & container monitoring is still very nascent, not a lot of (good) tools out there to protect (or attack) your infra
- Save your CISO's job by protecting against a moderately sophisticated attack

Q&A

https://github.com/nickswang/kubekit

Black Hat Sound Bytes

- Kubernetes is highly complex with many systems that attackers can use to hide.
- Container runtimes (docker sockets) aren’t the only attack surface, think of your external facing data plane.
- Kubernetes not currently as large a target as it should be (honeypots not being hit), but it could be! Kubekit can help your red and blue teams get ahead of the curve in detection and response!