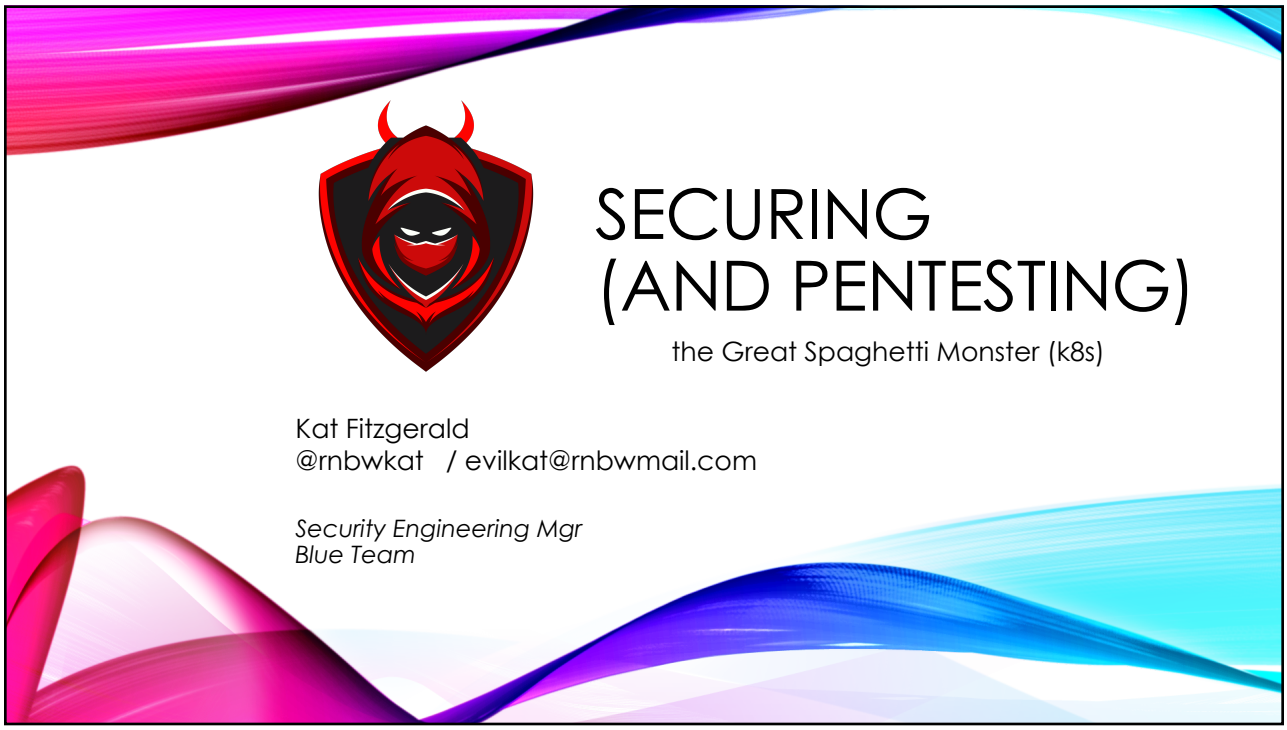


1

WHOAMI

$ whoami

Kat Fitzgerald (@rnbwkat or evilkat@rnbwmail.com)

- *CEO @BSidesChicago, 2019 COO @dianainitiative, CFP Chair @BSidesPGH, DefCon 3!*
- *Many years in Security, with an emphasis on Blue Teams, (former Purple), DevSecOps, IR.*
- *Based in ~~Pittsburgh~~ Kirkland, WA and a natural creature of winter, you can typically find me sipping Grand Mayan Extra Anejo whilst simultaneously defending my systems using OSS, magic spells and Dancing Flamingos.*
- *Honeypots & Refrigerators are a few of my favorite things!*

2

DISCLAIMER

- The views and opinions expressed in this presentation are my own and do not necessarily reflect the official policy or position of any current or previous employer. Examples of exploitations, coding and vulnerabilities discussed within this presentation are only examples and they should not be utilized in the real-world.

Those of you with an overwhelming fear of the unknown will be happy to learn that there is no hidden message revealed by reading this disclaimer backwards.

3

WHY WE ARE NOT HERE

- I won't solve all your k8s Security woes
- Neither will the person sitting next to you
- Common Sense went out the window decades ago

4

WHY WE ARE HERE

- Kubernetes is (still) new (and fun and crazy and fun)
- Containers are not new
  - (but they seem to be forgotten?)
- Security is for EVERYONE
- Common Sense is REQUIRED (disregard previous slide)

5

SOME SECURITY FACTS

- $114 B! (2018)
- Breaches are commonplace
- Security Appliances/software vulnerable
- Your Security Architecture is not unique

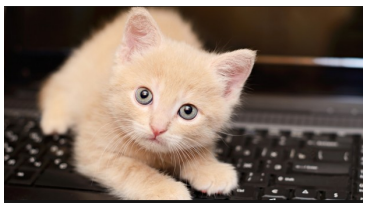
Instead of Brilliance, we have standardized mediocrity.
– John Strand, Offensive Countermeasures

6

WHAT IS A BREACH?

- Most breaches are NOT 0-day
- Most breaches are NOT "fancy"
- Most breaches don't come from "vulnerability scanners"

- Most breaches come from "Config Issues"
- A close 2nd - compromised credentials
- Trailing in 3rd - Over-Priv Users
- k8s = all 3!!



7

FEATURE

Capital One hack shows difficulty of defending against irrational cybercriminals

The motivation of the malicious actor who stole data of more than 100 million people was driven by emotional distress and did not follow traditional hacker patterns.

By Cynthia Brumfield

CSO | AUG 26, 2019 10:38 AM PDT

BUT…

CRYPTOCURRENCY JACKING —

Tesla cloud resources are hacked to run cryptocurrency-mining malware

Crooks find poorly secured access credentials, use them to install stealth miner.

DAN GOODIN - 2/20/2018, 2:21 PM

EQUIFAX
-HACKED-

8

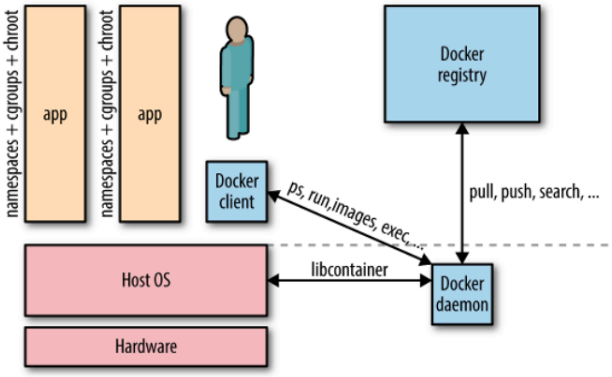
CONTAINERS – QUICK REVIEW

- Not secure by default
- A grouping of stuff
- Stuff = Resources
  - Resources are "namespaces"
  - Resources: processes, network, *users*, ports, IPC, etc
  - Resources are exploitable!
- Control Groups (cgroups) = limits
  - Important!

9

CONTAINERS - IMAGES

- *Your container*
- FS Snapshot
  - Who decides?
  - Stripped down
- Container Isolation
  - Really?



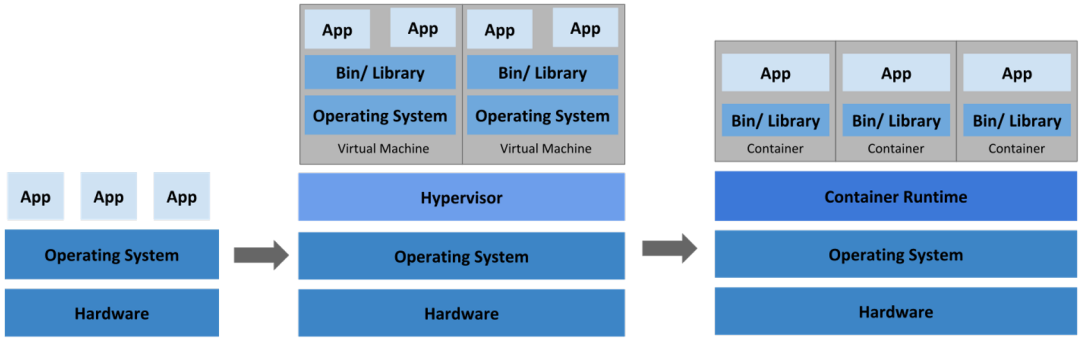
10

CONTAINERS – BEST PRACTICES

- Base Image – Secure Repo
  - CIS! (protections, ownership, etc)
- !root
- Visibility *(will talk more shortly)*
- Dominos
- NO new privs!
- Secrets Mgmt (DOH!)
- Stop allowing ssh
- Namespace LIMITS!
- DockerSlim - https://github.com/docker-slim/docker-slim
  - Apply SElinux and/or AppArmor
- Remember the Infra! *Ding! Ding! Ding! (reminder forthcoming)*

11

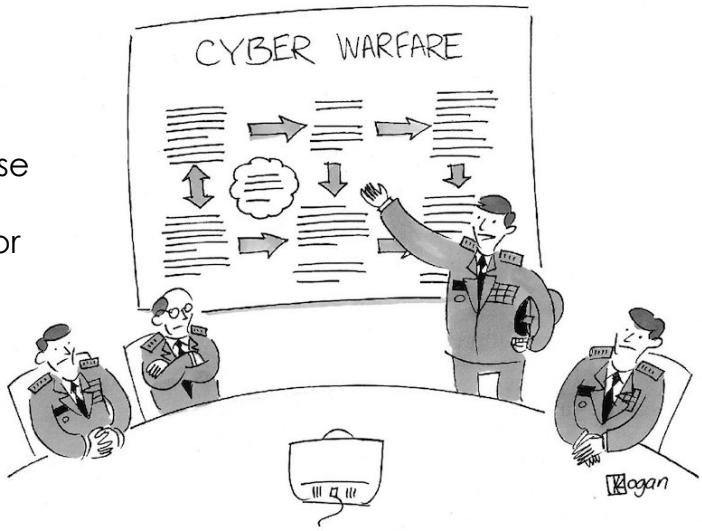
A PICTURE?



12

A Good (great?) Defense

Bring on the Orchestrator



"First, we inundate them with quarterly Kubernetes releases."

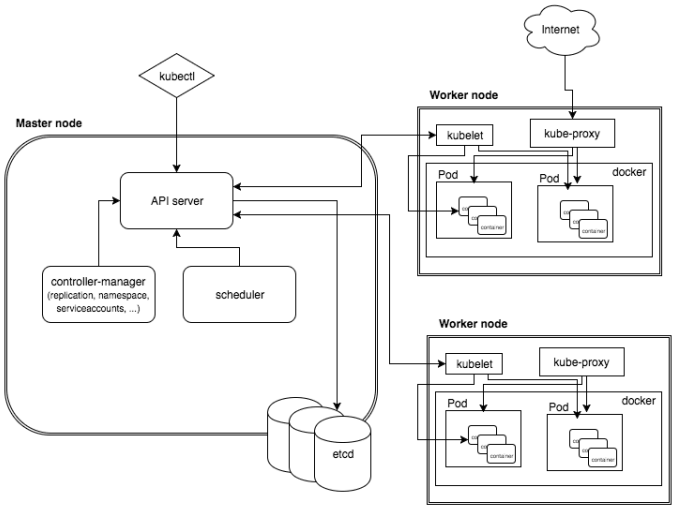
13

K8S 101

- *Master Node*
  - The node which manages the k8s worker node cluster and the deployment of pods on nodes.
- *Worker Node*
  - These nodes typically run the application containers and other k8s components such as agents and proxies.
- *Pods*
  - A unit of deployment and addressability in k8s. A pod has its own IP address and can contain one or more containers. (think subnets/vlans/zones)
- *Services*
  - A service functions as a proxy to its underlying pods and requests can be load balanced across replicated pods.
- *Main Components*
  - Key components which are used to manage a k8s cluster include the API Server, Kubelet, and etcd.

14

K8s Architecture



15

NOT PERFECT (TELL ME IT ISN'T SO!?!?!)

- K8s is not perfect
- https://blog.rapid7.com/2018/06/27/analyzing-the-kubernetes-hack-backdooring-through-the-kubelet-api/

- *By default, requests to the kubelet's HTTPS endpoint that are not rejected by other configured authentication methods are treated as anonymous requests and given a username of system:anonymous and a group of system:unauthenticated.*

  - https://kubernetes.io/docs/reference/command-line-tools-reference/kubelet-authentication-authorization/
  - https://medium.com/handy-tech/analysis-of-a-kubernetes-hack-backdooring-through-kubelet-823be5c3d67c

16

WHEN COMMON SENSE WINS!
(THREAT MODELING)

- TLS Everywhere!!!!!!!!!!!!!!!!!! (too many !'s ?)
- Harden the Infrastructure ← *Remember, I said I would mention this again.*
- Enable RBAC with Least Privilege
  - Disable ABAC, and Monitor Logs
  - https://wazuh.com/blog/auditing-kubernetes-with-wazuh/
  - kubectl get rolebinding --all-namespaces ← *trust but verify!*
- Use Third Party Auth for API Server
- Separate and Firewall your etcd Cluster
- Rotate Encryption Keys
  - Use Vault
- Use Linux Security Features and PodSecurityPolicies
  - AppArmor/SElinux

17

RISK(Y) BUSINESS

- kubectl get secrets
  - https://github.com/cyberark/kubernetes-rbac-audit
  - https://github.com/cyberark/kubiscan
- Git[hub | lab]
- Image repos/registries
  - Libraries!
- Applications
  - Pods
- Shodan (yes, even for k8s https://shodan.io)

18

CONFIGS

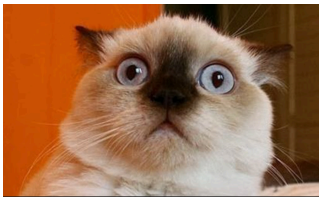
- Remember - Most breaches come from "Config Issues"
- CIS Hardening (Center for Internet Security)
  - https://www.cisecurity.org/cis-benchmarks/
  - Docker - https://www.cisecurity.org/benchmark/docker/
  - K8s - https://www.cisecurity.org/benchmark/kubernetes/
  - You get the idea…

But what about Automation?

19

BUT WE USE JENKINS/CI/CD

- But we use Jenkins (some kind of CI/CD integration)?
- Anchore Container Image Scanner Plugin
  - https://plugins.jenkins.io/anchore-container-scanner
  - https://wiki.jenkins.io/display/JENKINS/Anchore+Container+Image+Scanner+Plugin
  - Analyze, Inspect and perform security scans against images



20

SOME TOOLS FOR YOU

- Kube-bench - https://github.com/aquasecurity/kube-bench
  - Runs against the CIS k8s benchmark
  - Run it against Master or Node
- KubeAudit - https://github.com/Shopify/kubeaudit
  - More granular that kubesec
  - Audit the cluster against common controls
  - Cluster, Local and Manifest modes

21

QUICK EXAMPLES

```
INFO[0000] Not running inside cluster, using local config
ERRO[0001] AllowPrivilegeEscalation not set which allows privilege escalation, please set to false  Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] ReadOnlyRootFilesystem not set which results in a writable rootFS, please set to true  Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] RunAsNonRoot is not set in ContainerSecurityContext, which results in root user being allowed!  Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
WARN[0001] serviceAccount is a deprecated alias for ServiceAccountName, use that one instead  DSA=fission-svc KubeType=daemonSet Name=logger Namespace=fission SA=fission-svc
WARN[0001] Privileged defaults to false, which results in non privileged, which is okay.  Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] Capability not dropped                        CapName=SETFCAP Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] Capability not dropped                        CapName=SETGID Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] Capability not dropped                        CapName=SETPCAP Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] Capability not dropped                        CapName=SETUID Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] Capability not dropped                        CapName=SYS_CHROOT Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
WARN[0001] Resource limit not set, please set it!        KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] AppArmor annotation missing.                  Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] Seccomp annotation missing.                   Container=fluentd KubeType=daemonSet Name=logger Namespace=fission
ERRO[0001] AllowPrivilegeEscalation not set which allows privilege escalation, please set to false  Container=calico-node KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] AllowPrivilegeEscalation not set which allows privilege escalation, please set to false  Container=install-cni KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] ReadOnlyRootFilesystem not set which results in a writable rootFS, please set to true  Container=calico-node KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] ReadOnlyRootFilesystem not set which results in a writable rootFS, please set to true  Container=install-cni KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] RunAsNonRoot is not set in ContainerSecurityContext, which results in root user being allowed!  Container=calico-node KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] RunAsNonRoot is not set in ContainerSecurityContext, which results in root user being allowed!  Container=install-cni KubeType=daemonSet Name=calico-node Namespace=kube-system
WARN[0001] serviceAccount is a deprecated alias for ServiceAccountName, use that one instead  DSA=calico KubeType=daemonSet Name=calico-node Namespace=kube-system SA=calico
ERRO[0001] Privileged set to true! Please change it to false!  Container=calico-node KubeType=daemonSet Name=calico-node Namespace=kube-system
WARN[0001] Privileged defaults to false, which results in non privileged, which is okay.  Container=install-cni KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] Capability not dropped                        CapName=AUDIT_WRITE Container=calico-node KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] Capability not dropped                        CapName=SYS_CHROOT Container=install-cni KubeType=daemonSet Name=calico-node Namespace=kube-system
WARN[0001] Resource limit not set, please set it!        KubeType=daemonSet Name=calico-node Namespace=kube-system
WARN[0001] Resource limit not set, please set it!        KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] AppArmor annotation missing.                  Container=calico-node KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] AppArmor annotation missing.                  Container=install-cni KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] Seccomp annotation missing.                   Container=calico-node KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] Seccomp annotation missing.                   Container=install-cni KubeType=daemonSet Name=calico-node Namespace=kube-system
ERRO[0001] AllowPrivilegeEscalation not set which allows privilege escalation, please set to false  Container=fluentd KubeType=daemonSet Name=fluentd Namespace=logging
ERRO[0001] ReadOnlyRootFilesystem not set which results in a writable rootFS, please set to true  Container=fluentd KubeType=daemonSet Name=fluentd Namespace=logging
ERRO[0001] RunAsNonRoot is not set in ContainerSecurityContext, which results in root user being allowed!  Container=fluentd KubeType=daemonSet Name=fluentd Namespace=logging
WARN[0001] serviceAccount is a deprecated alias for ServiceAccountName, use that one instead  DSA=fluentd-sa KubeType=daemonSet Name=fluentd Namespace=logging SA=fluentd-sa
WARN[0001] Privileged defaults to false, which results in non privileged, which is okay.  Container=fluentd KubeType=daemonSet Name=fluentd Namespace=logging
```

22

MORE TOOLS

- Clair - https://github.com/coreos/clair
  - Static analysis for application containers
- Klar - https://github.com/optiopay/klar
  - Integrates Clair and Docker Registry
- Falco - https://falco.org/
  - Behavioral Activity Monitoring
  - Detects "suspicious" things
  - Full Container support
- Kube-hunter - https://github.com/aquasecurity/kube-hunter

23

READY SET ATTACK!

Port	Process	Description
443/TCP	kube-apiserver	Kuberntes API port
2379/TCP	etcd	
6666/TCP	etcd	etcd
4194/TCP	cAdvisor	Contrainer metrics
6443/TCP	kube-apiserver	Kubernetes API port
8443/TCP	kube-apiserver	Minikube API port
8080/TCP	kube-apiserver	Insecure API port
10250/TCP	kubelet	HTTPS API which allows full node access
10255/TCP	kubelet	Unauthenticated read-only HTTP port: pods, runningpods and node state
10256/TCP	kube-proxy	Kube Proxy health check server
9099/TCP	calico-felix	Health check server for Calico
6782-4/TCP	weave	Metrics and endpoints

24

I ARE HACKER
SEE ME HACK

- Checking Access to the API Server
- Checking for ETCD Access
- Checking Kubelet (Read Only Port)
- Container Vulns / Exposure
  - Sensitive Files
  - Remember falco?
- Kernel Exploits
  - Priv Escalation
- Vulnerable Apps

25

FINAL THREAT MODEL

- Think of your entire CI/CD pipeline

1. Workload (pods) Security
2. Container Security
3. File Storage Security
4. Network Security
5. Application Security

26

REMEMBER!!!!

- Most breaches are NOT 0-day
- Most breaches are NOT "fancy"
- Most breaches don't come from "vulnerability scanners"

- Most breaches come from "Config Issues"
- A close 2nd - compromised credentials
- Trailing in 3rd - Over-Priv Users

27

KEY TAKE AWAYS

- Common Sense – *FTW!*
  - *Stop overthinking*
- The Basics ← *Ding ding ding*!
  - *Stop the Fancy*
- Secure the Environment – CIS
  - RBAC, Infra, Pods, Containers, Network, Apps
- Threat Modeling
- Access Controls
  - Did I mention RBAC?
- PATCH!
- Logging & Auditing/Monitoring!!
  - https://wazuh.com/blog/auditing-kubernetes-with-wazuh/

28

Kat Fitzgerald
evilkat@rnbwmail.com

@rnbwkat

29