Incident Response in containerized or ephemeral environments

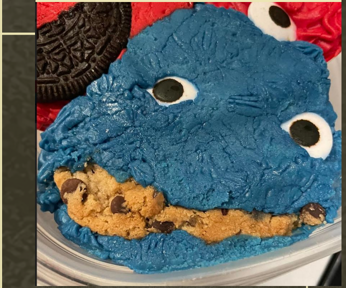
David Mitchell
&
Adrian Wood

Presenters

David Mitchell

@digish0
https://keybase.io/digisho

Adrian Wood



@whitehacksec
https://keybase.io/threlfall

01

Threat Landscape

Why this talk matters

02

Problem Space

Complexities in container and ephemeral environments

03

Preparedness

Preparation is key to your response

04

Execution

Scenarios and Forensics

05

Tying it all together

Using eBPF and other technologies

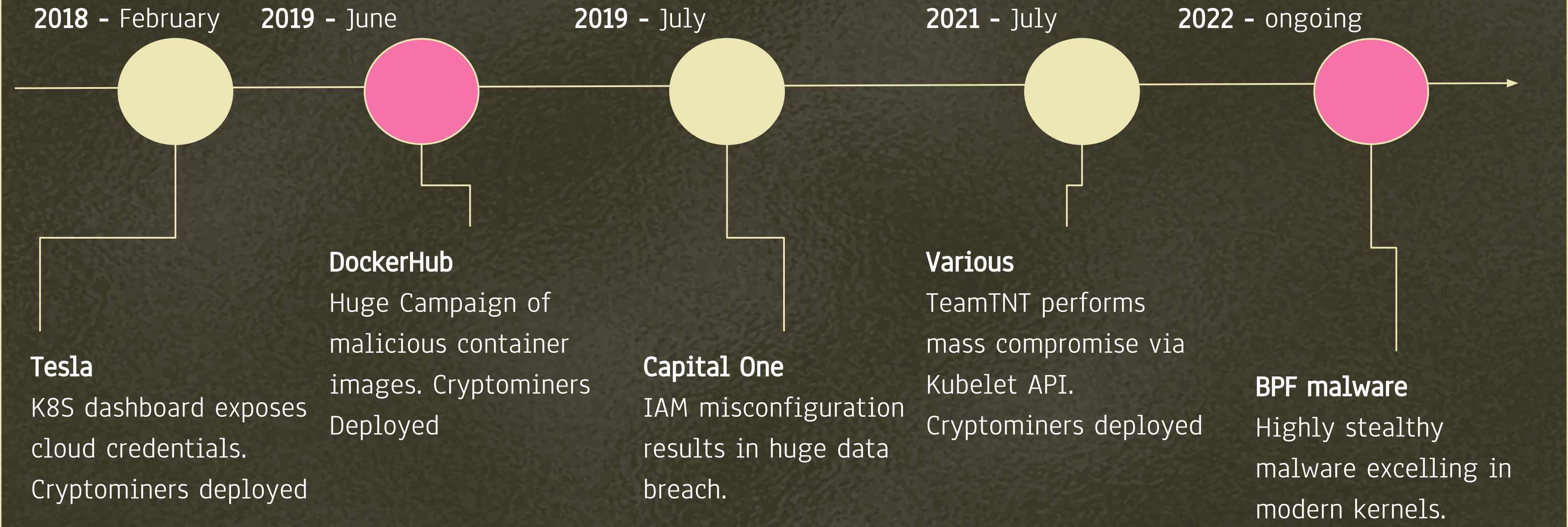
06

Conclusion && Questions

You can describe the topic of the section here

01 Threat Landscape

Not exhaustive



2018 - February

Tesla
K8S dashboard exposes cloud credentials. Cryptominers deployed

2019 - June

DockerHub
Huge Campaign of malicious container images. Cryptominers Deployed

2019 - July

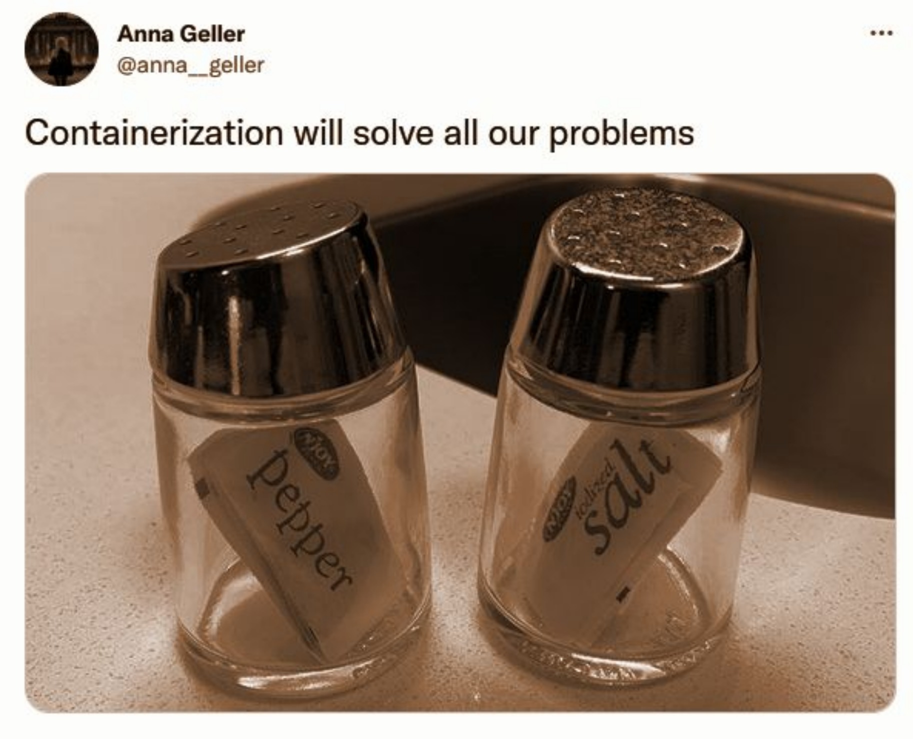
Capital One
IAM misconfiguration results in huge data breach.

2021 - July

Various
TeamTNT performs mass compromise via Kubelet API. Cryptominers deployed

2022 - ongoing

BPF malware
Highly stealthy malware excelling in modern kernels.



Anna Geller
@anna__geller

Containerization will solve all our problems

00 Intro Info | Quick Context

Containers are:

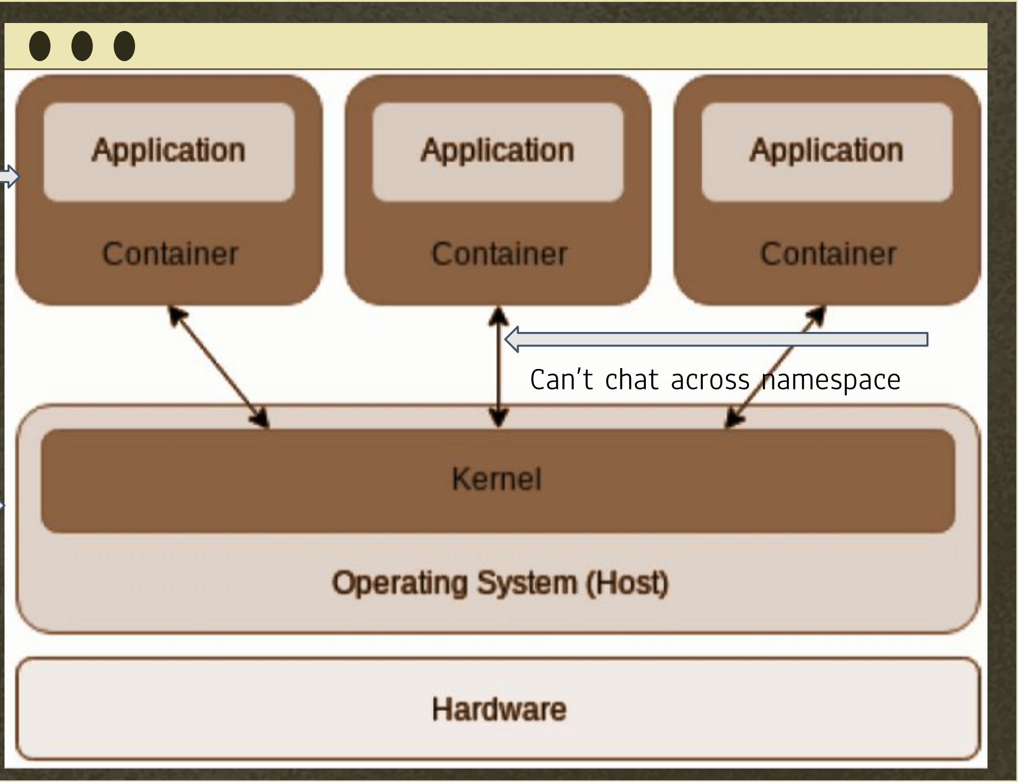
“processes, born from tarballs, anchored to namespaces, controlled by cgroups.”

Just a tarball

with some metadata

cgroups dictate what process resources

can be leveraged by the container



Can’t chat across namespace

00 Intro Info | Quick Context

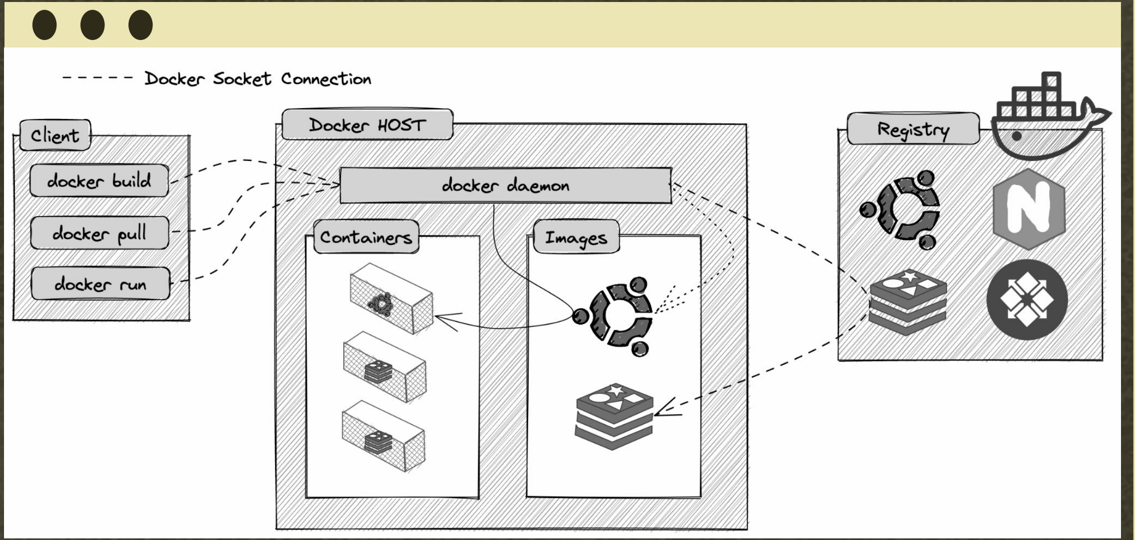
Docker is:

Simply a way of managing a lot of

these processes, in easy, portable

configurations.

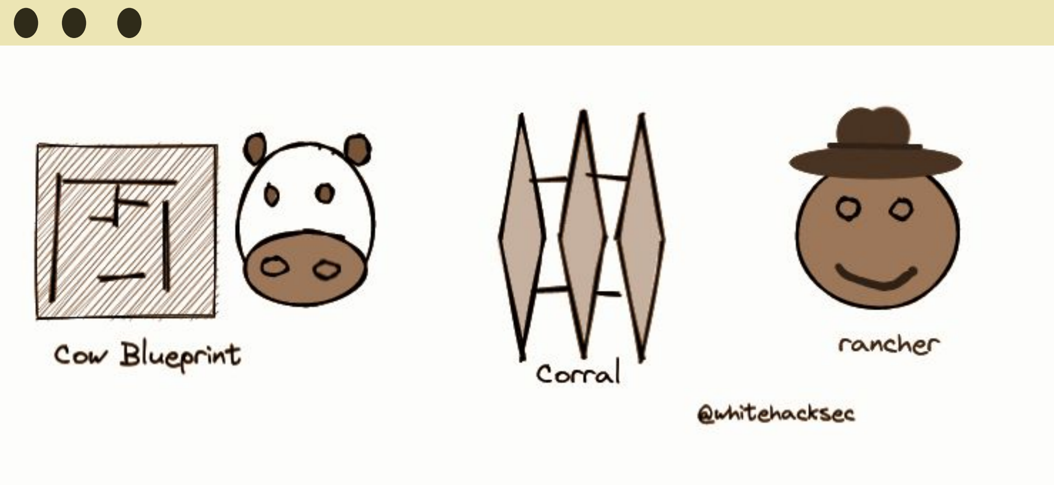
“Cattle, not pets”



00 Intro Info | Quick Context

Kubernetes is:

A rancher, ensuring that their fleet of cows have the appropriate resources, moving them and managing them.



Ø2 Problem Space

Ø1

Complexity

Of tracing, of management, of identities

Ø2

Logging

Additional sources, huge volume

Ø3

Attack Surface

Preparing for the change in attack surface

Ø4

Migration

To ephemeral and cloud compute changes IR strategies

Ø5

Identity Management

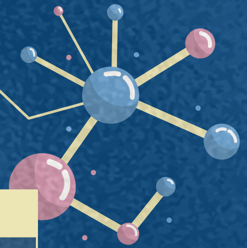
Complex layers of identity management

Ø6

Ephemeral Instrumentation

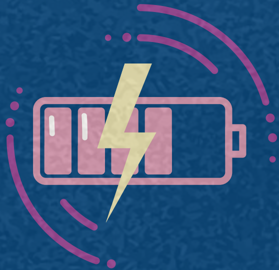
Is difficult, moreso when you're on shared hardware

03 Preparation



“There is no shorter road to defeat than by entering a war with inadequate preparation.””

—Charles Lindbergh



03 Preparation

Two primary areas:

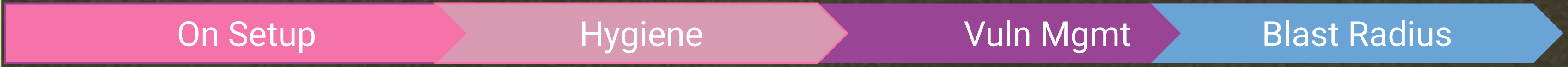
-Prevention

-Collection

03 Preparation | Prevention

- Minimal (hardened) OS images
- Audit Logging
- **CI/CD Controls**
- Verify Binaries
- Tight IAM
- Private IP's on nodes

- Limit Pod Identities
- Use a service mesh
- Protect Secrets
- PodSecurity Admission controller



On Setup

Hygiene

Vuln Mgmt

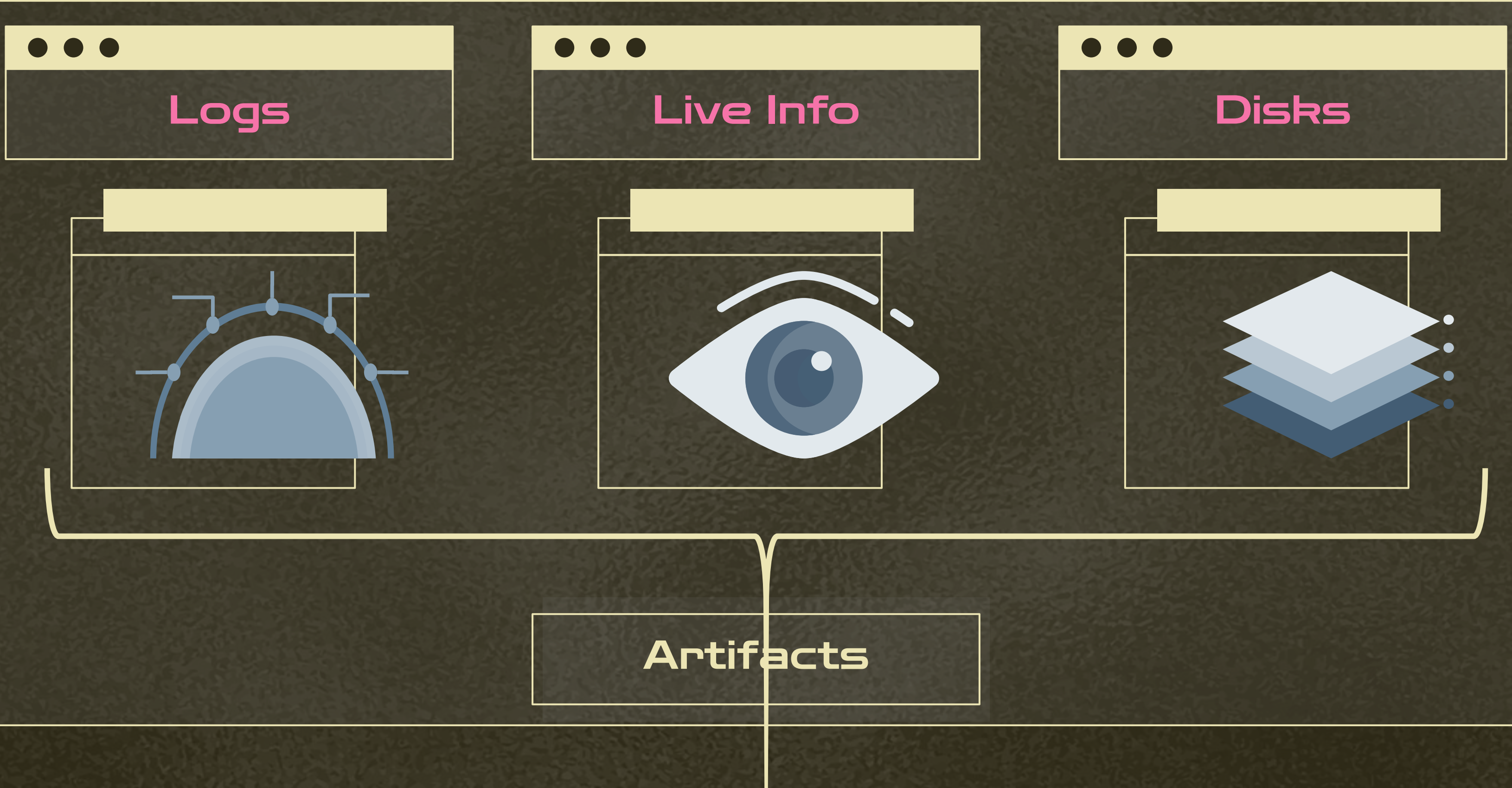
Blast Radius

- **Create an IR project**
- Restrict access to kubectl
- Use RBAC
- Use Namespaces
- (bootstrap) TLS
- Network Policies
- IR Playbooks

- Scan for known vulns
- **Sandboxing/Quarantine pattern**
- Disable default tokens
- **Security tools on host**

03 Preparation | Collection

Build a Story



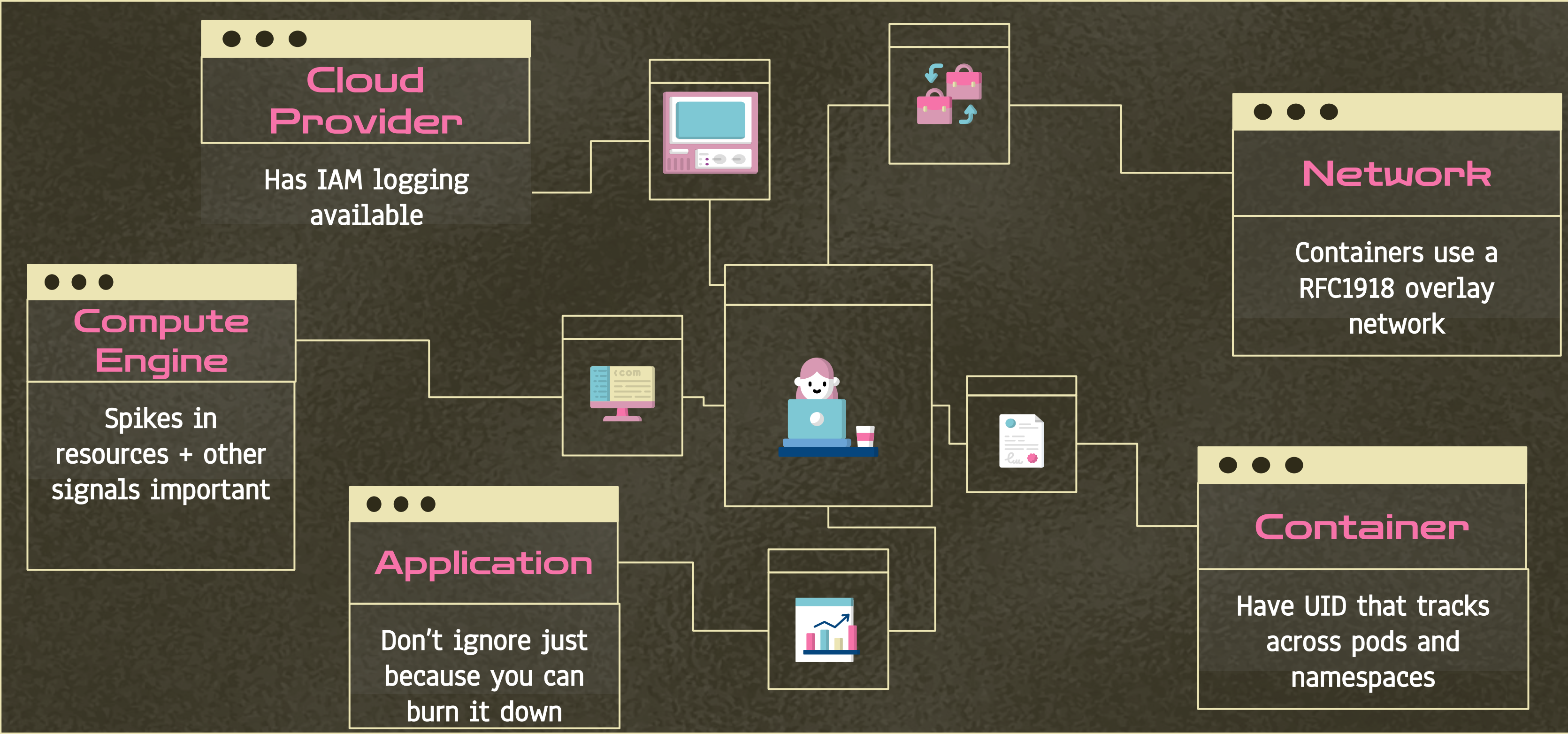
Logs

Live Info

Disks

Artifacts

03 Preparation | Collection - Logs



Cloud Provider

Has IAM logging available

Compute Engine

Spikes in resources + other signals important

Application

Don't ignore just because you can burn it down

Network

Containers use a RFC1918 overlay network

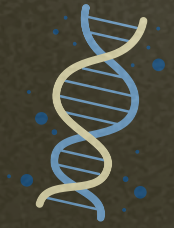
Container

Have UID that tracks across pods and namespaces

03 Preparation | Collection -- Live Info

Client
Agents

Container
Sidecars



Awareness

What is happening on the system?

Opsec

How will you get info without logging in?

Reality

Dealing with Multiple Infections

03 Preparation | Collection - Disks

Traditional

Snapping a disk for offline analysis is easy

Cloud

Cloud APIs make it easy to take a snapshot

Container

No Container Snapshot Mechanism (manual)

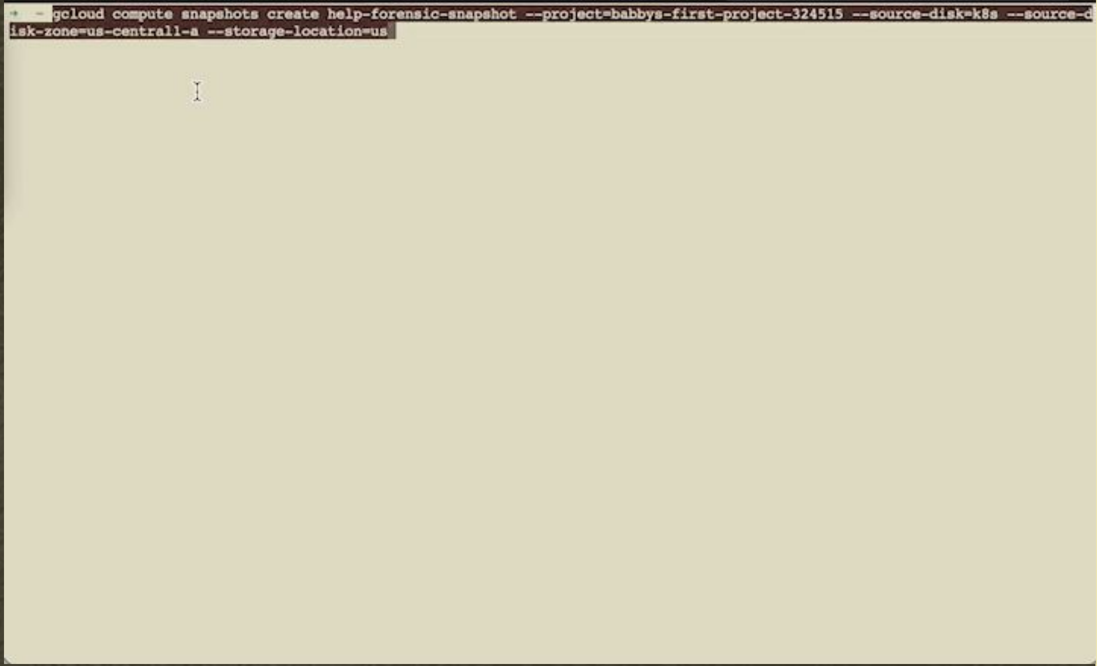
Do you have a strategy to take multiple snapshots? Can you diff off known good?

03 Preparation | Collection -- Snapshotting

Snapshot Permissions

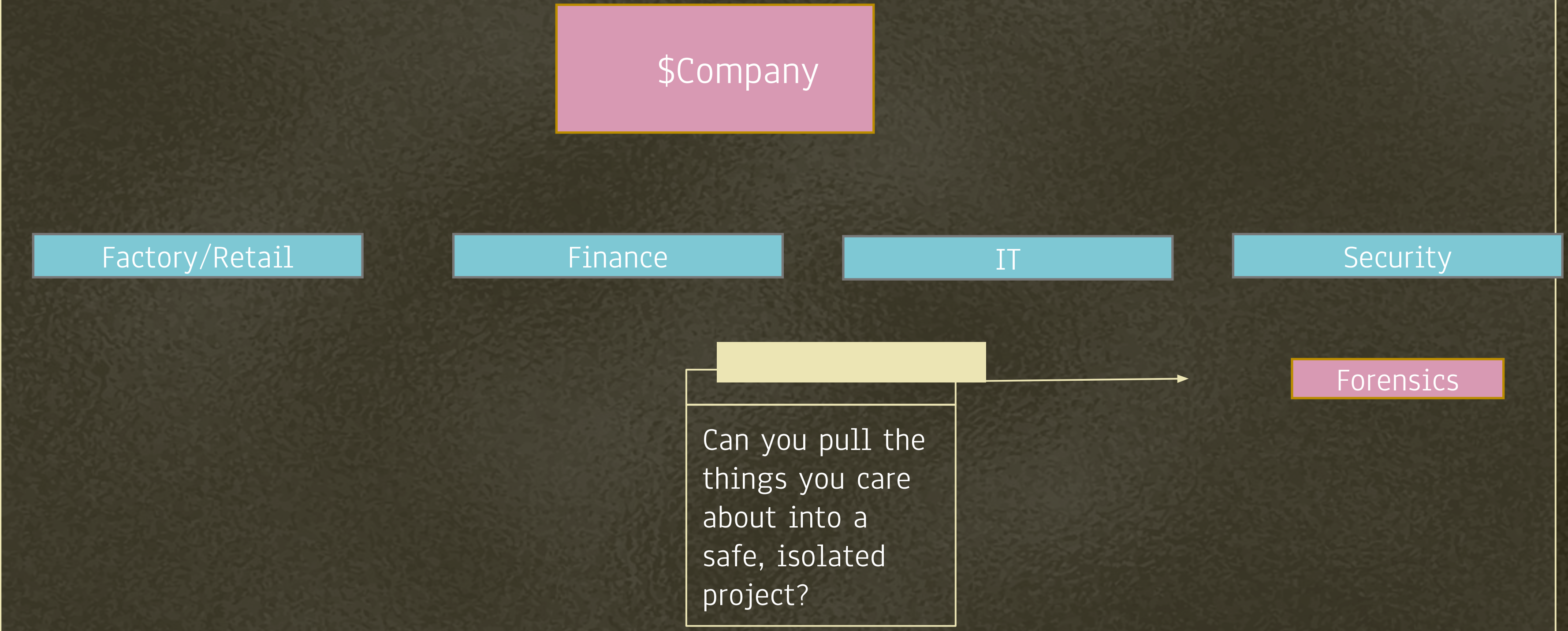
Do you have permissions to snapshot across the fleet?

How are the permissions managed, accessed and audited?



```
→  ~ gcloud compute snapshots create help-forensic-snapshot --project=babbys-first-project-324515 --source-disk=k8s --source-d
isk-zone=us-central1-a --storage-location=us
```

03 Preparation | Collection - where?



$Company

Factory/Retail

Finance

IT

Security

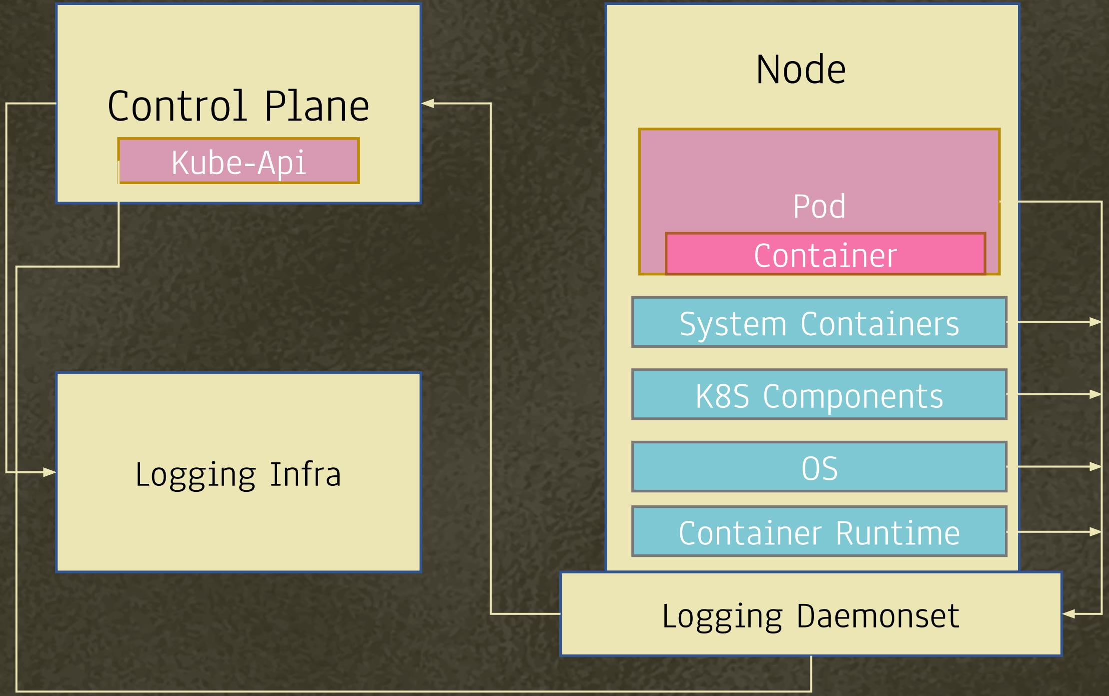
Forensics

Can you pull the things you care about into a safe, isolated project?

K8s Log$

There's alot.

03 Preparation | Collection -- K8S



03 Preparation | Collection -- K8S Audit Logging

None

No Events Logged

Request+Metadata

Request BODY plus Metadata

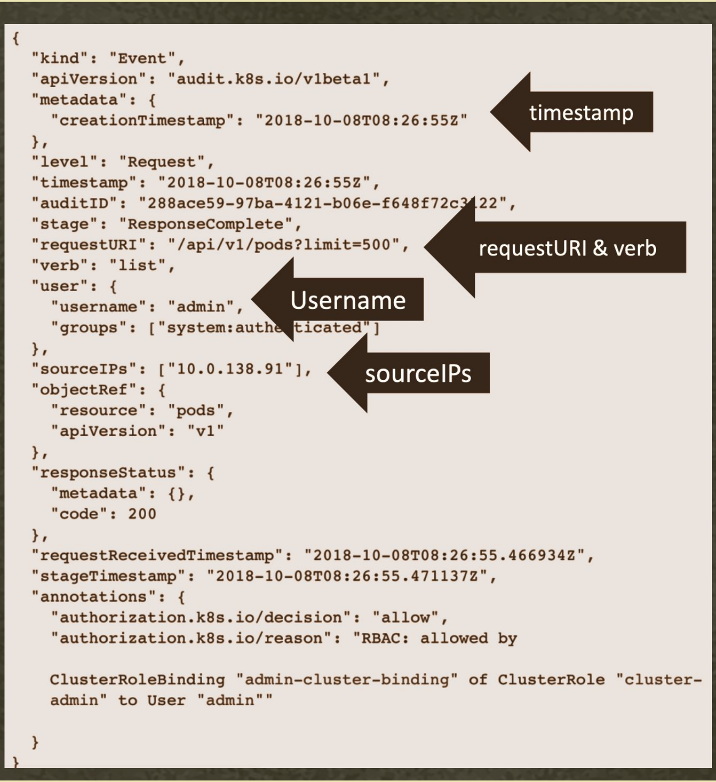
Metadata

Request Metadata Only

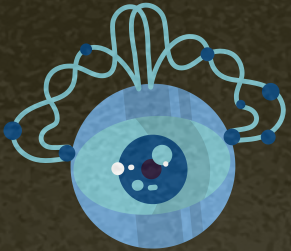
RequestResponse

Full Request and Response + Metadata

03 Preparation | Collection -- K8S Audit Logging

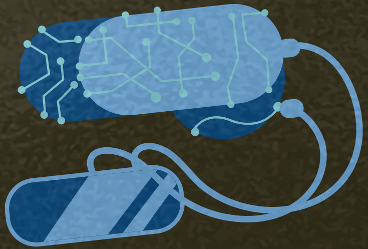


```
{
  "kind": "Event",
  "apiVersion": "audit.k8s.io/v1beta1",
  "metadata": {
    "creationTimestamp": "2018-10-08T08:26:55Z"
  },
  "level": "Request",
  "timestamp": "2018-10-08T08:26:55Z",
  "auditID": "288ace59-97ba-4121-b06e-f648f72c3122",
  "stage": "ResponseComplete",
  "requestURI": "/api/v1/pods?limit=500",
  "verb": "list",
  "user": {
    "username": "admin",
    "groups": ["system:authenticated"]
  },
  "sourceIPs": ["10.0.138.91"],
  "objectRef": {
    "resource": "pods",
    "apiVersion": "v1"
  },
  "responseStatus": {
    "metadata": {},
    "code": 200
  },
  "requestReceivedTimestamp": "2018-10-08T08:26:55.466934Z",
  "stageTimestamp": "2018-10-08T08:26:55.471137Z",
  "annotations": {
    "authorization.k8s.io/decision": "allow",
    "authorization.k8s.io/reason": "RBAC: allowed by

    ClusterRoleBinding "admin-cluster-binding" of ClusterRole "cluster-
    admin" to User "admin""

  }
}
```



Container Forensics

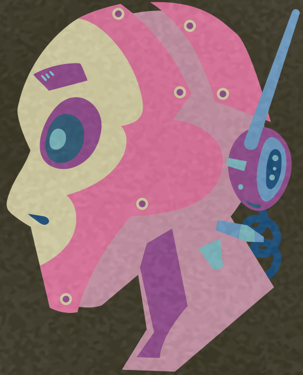
Despite the hype it is actually necessary

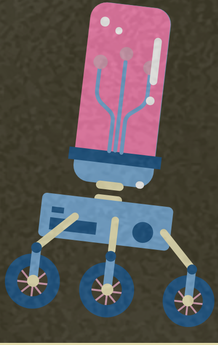


04 Execution | Forensics - General Notes

Don't log in

Stay off the container.





Wiping?

Lots of people tout that the benefit of containers is wiping and starting over...

04 Execution | Forensic Strategies

Response	Condition	Action	Reason
Isolate	No Data Exfil / type within classification tolerance	Cordon workload	Observe attacker/discovery
Pause		Stop running processes	Cryptomining
Restart		Kill and restart	Gets rid of attacker, temporarily Rolling out a new patched image
Kill	Data exfil	Kill workloads	Prevent data leakage/loss

04 Execution | Forensic strategies - Isolation

1. Apply a label to node and pod (e.g. IRteam) denoting it is under investigation

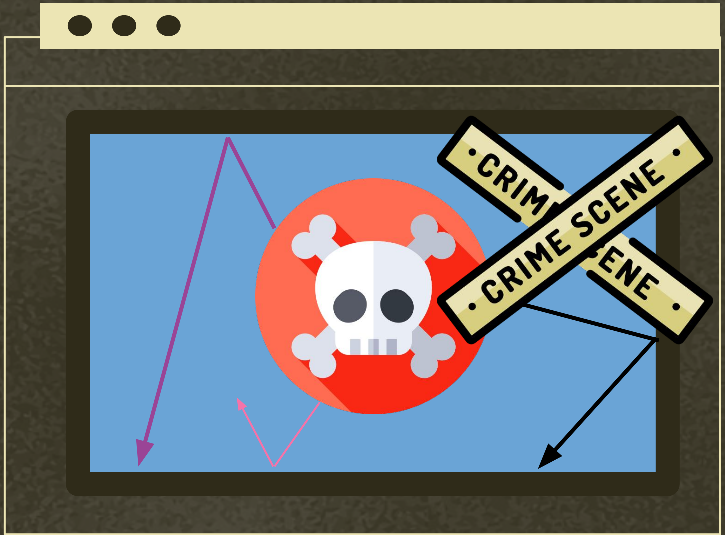
2. Revoke security credentials assigned to pod

3.Create network policy to isolate traffic ingress egress traffic from pod

4.Cordon the node

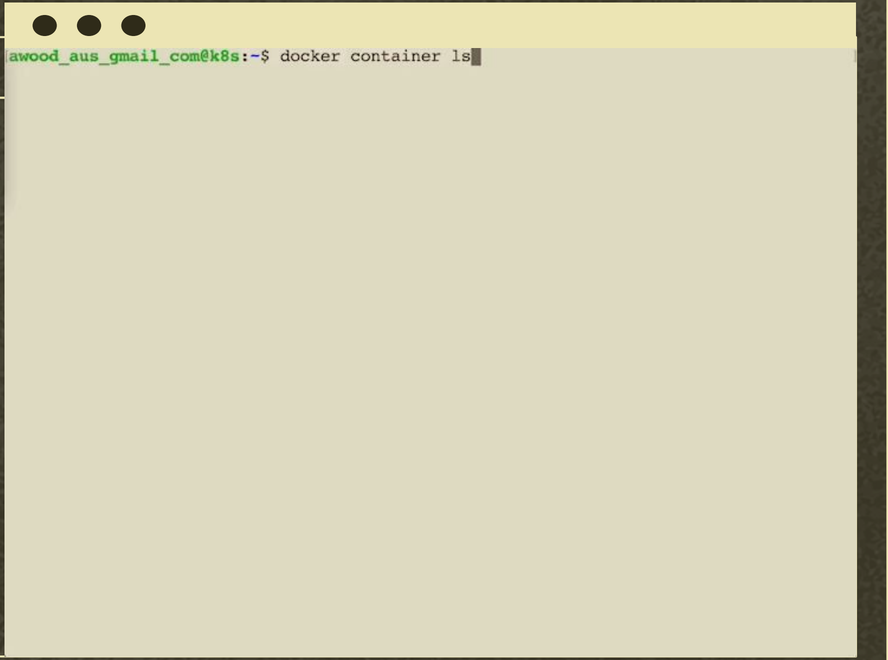
5.Drain other workloads from it

6.Capture volatile artifacts ASAP



04 Execution | Forensic strategies - Isolation

```
$ kubectl cordon
```



```
awood_aus_gmail_com@k8s:~$ docker container ls
```

04 Execution | Forensic strategies - Pause

- No easy way to do this in K8S except through resource constraints

```
$ docker pause
```

- Usually done to preserve container that is consuming lots of resources (cryptominer)
- Execution pausing of processes also takes place temporarily while a snapshot is being taken of container or VM state

04 Execution | Forensic strategies - Restart

- Unless you're restarting to apply a patch, doesn't fix your problems.
- Attacker will just come back
- Attacker may still be in environment somewhere else.
- May be told/ordered to do this to get the business back online

04 Execution | Forensic strategies - Kill

As a last resort, you may wish to kill.
You’ll need to stop all processes instantly, without restart, in cases such as ongoing data loss, privilege escalations and lack of visibility.

```
$ docker stop (sigterm & sigkill
after 10 secs)

$ docker kill (sigkill)

$ kubectl delete
```

Fancy detection technologies

05 Tying it all together | eBPF

Latest and greatest, pretty revolutionary

Lets you extend the kernel without modifying source code or making kernel modules (all of this is very hard)

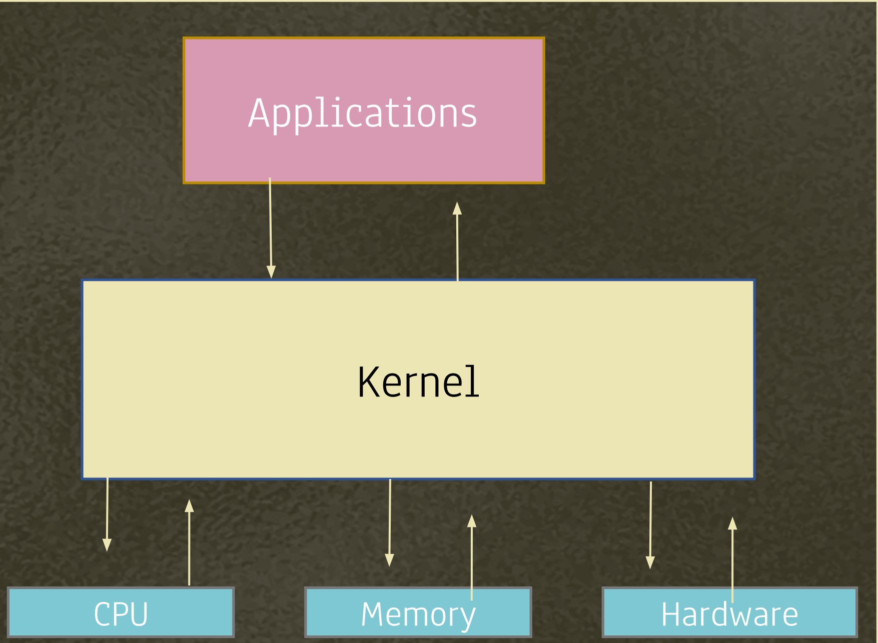
The kernel is the perfect place for observability functionality, if you can clear the VERY high bar for entry.

05 Tying it all together | eBPF - ELI 5

You aren’t a chef, you can’t use or do the things in the kitchen

The kitchen equipment like the stove (hard drives and computer bits) you don’t know how to operate

It would be nice to have something at hand that can go into the kitchen and look around on your behalf. eBPF.

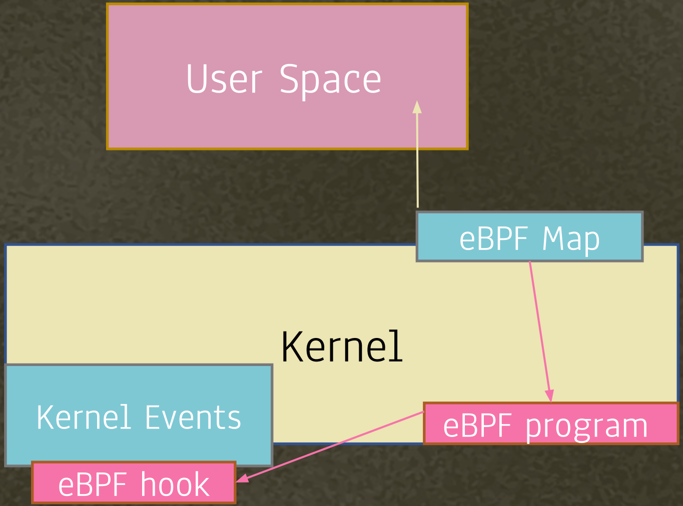


05 Tying it all together | eBPF

By using system hooks, we can monitor for system calls, network events, or anything, triggering the program to report this back to the user space.

It can also be used for rootkits and malware itself!

Amazing networking observability and security functionality.

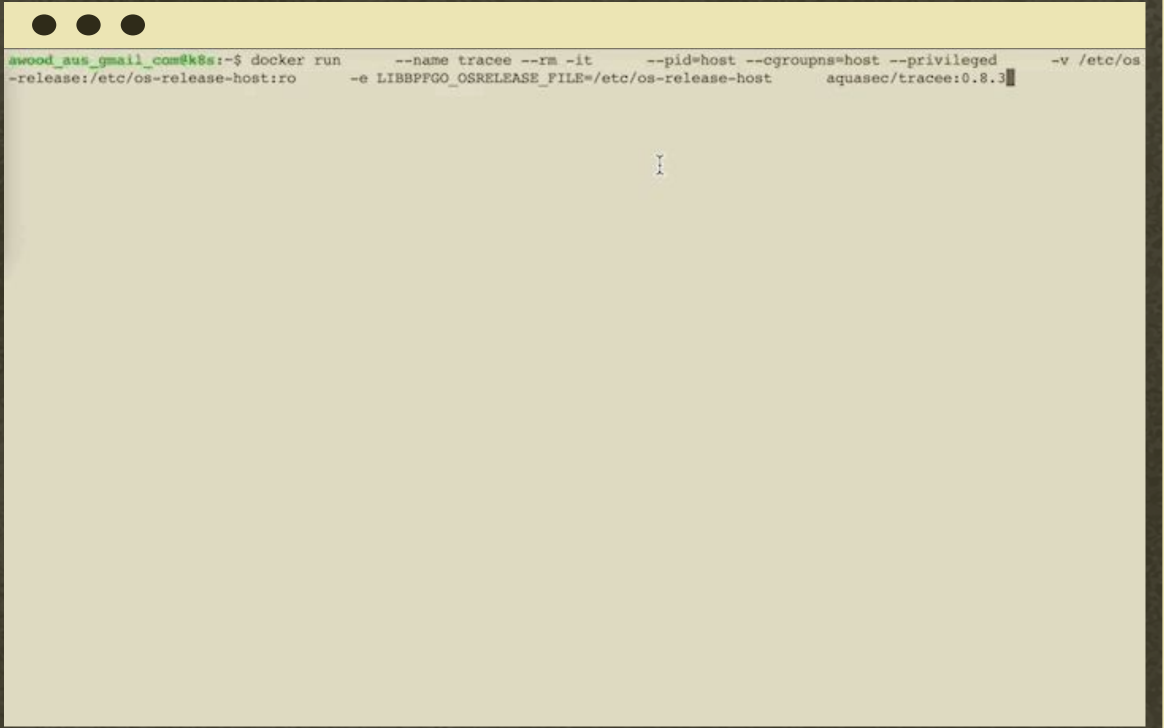


Add your own observing chef to the kitchen with little effort; they've been vetted and will act right.

05 Tying it all together | eBPF

eBPF tooling gives incredibly powerful views into system activity

& the rise of eBPF malware, eBPF detections are a must.



```
awood_aus_gmail_com@k8s:~$ docker run      --name tracee --rm -it      --pid=host --cgroupns=host --privileged      -v /etc/os
-release:/etc/os-release-host:ro      -e LIBBPFGO_OSRELEASE_FILE=/etc/os-release-host      aquasec/tracee:0.8.3
```

05 Tying it all together | Machine Learning

There are (now)some working use cases, some of which aren’t complete bullshit, strong points:

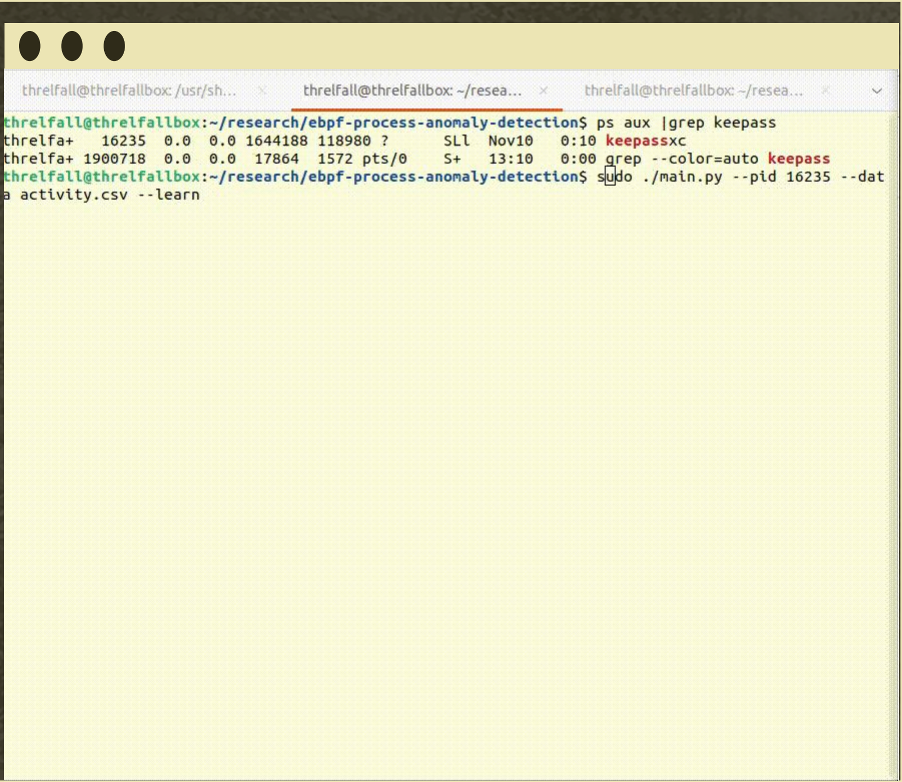
- Behavioral Profiling
- Anomaly Detection
- Reversing

eBPF pairs well with machine learning technologies, even for unsupervised learning:

05 Tying it all together | ML Tools - Anomalies

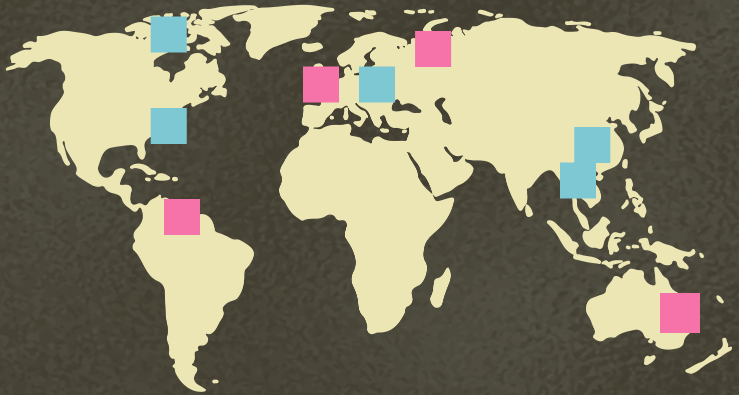
Great for:

- Detections
- Refining RASPS
- Research


threlfall@threlfallbox: /usr/sh...
threlfall@threlfallbox: ~/resea...
threlfall@threlfallbox: ~/resea...

```
threlfall@threlfallbox:~/research/ebpf-process-anomaly-detection$ ps aux |grep keepass
threlfa+   16235  0.0  0.0 1644188 118980 ?       SLl  Nov10   0:10 keepassxc
threlfa+ 1900718  0.0  0.0  17864  1572 pts/0    S+   13:10   0:00 grep --color=auto keepass
threlfall@threlfallbox:~/research/ebpf-process-anomaly-detection$ sudo ./main.py --pid 16235 --dat
a activity.csv --learn
```

05 Tying it all together | eBPF - Malware



eBPF Arms Race

eBPF malware is very hard to deal with, without eBPF.

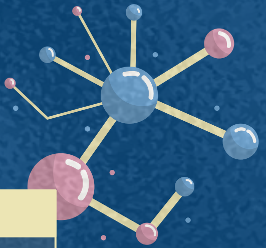
-Fileless malware

-More stable than a ROP chain

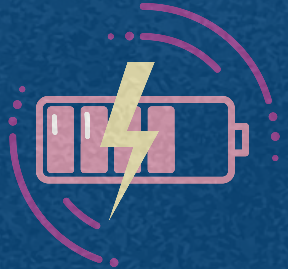
Bvp47 Malware

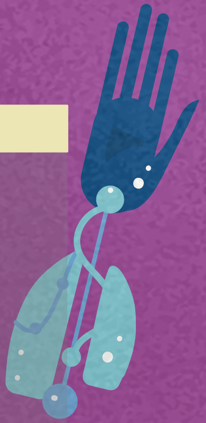
287 targets, 45 countries, years and years undetected.

But you only have 9 dots??



> “Don’t let the first time you go into battle be the first time you get punched in the face. Punch yourself in the face ahead of time. Oh, and have a plan.”
>
> —PRES. ABRAHAM WESTINGTON





THANK YOU

Questions?

Labs and resources:
https://github.com/lockfale/Malicious_Containers_Workshop

THANKS!

CREDITS: This presentation template was created by **Slidesgo,** including icons by **Flaticon,** and infographics & images by **Freepik**

Please keep this slide as attribution