Sysdig

Container Security Workshop

CfgMgmt-2019

Miguel Julián @miky_kun

Julio García @papajulio

https://github.com/figarocorso/security-workshop

<Spoiler Alert>

- Intro: who are we and what are we doing here?
  - QA @ Sysdig
- Container security best practices
- Kubernetes platform security features
  - Run a few cases of use
- Image scanning
  - Scan a couple of images
  - Jenkins integration live demo
- Runtime security and forensics
- Deploying the full set of open source tools
  - Full demo with Sysdig Falco integrated with Nats + Kubeless

Few years ago...

First approach to docker

- It was told to be easy… until something goes wrong
  - How do I ssh there?
  - It’s dead, so I cannot docker logs…

> I’m an analog man, trapped in a digital world
> Just an old school son of <beep> in this digital hell
> Don't need no wifi, just want my hi-fi
> Computers crashing, I want to smash 'em
> I'm an analog man, dying in a digital world
>
> Accept - Analog Man

Nowadays...

No stop signs
Speed limit
Nobody's gonna slow me down
Like a wheel
Gonna spin it
Nobody's gonna mess me around
Hey, Satan
Payin' my dues
Playin' in a rockin' band
Hey, mamma
Look at me
I'm on the way to the promised land
I'm on the highway to hell

AC/DC - Highway to Hell

Container security best practices

Resource usage

- Isolated does not mean they cannot disturb others
- Our family has grown bigger with containers (vs. virtualization)
  - Thus we have to share better resources
- DoS because of bug or malware
- Many resources to share
  - CPU, RAM, storage, I/O, ...
  - But also file descriptors, user IDs, directory entries, ..
    - https://sysdig.com/blog/container-isolation-gone-wrong/

docker run -it --memory=2G --memory-swap=3G ubuntu bash

Dockerfile

- COPY >> ADD
- Secrets never in Dockerfile
  - Nor at env variables, better use external tools like vaultproject
- Do not forward privileged ports (22!)
- Force user directive, avoid using root user
- Avoid unnecessary packages
  - Also clear as much as you can (keep images light)

Privileges

- Enforce mandatory access control to prevent non desire access
  - seccomp, selinux, apparmor
- Create a non-root user
  - Remove setuid and setgui permission
- Do not --userns=host (better isolate container users)
- --security-opt=no-new-privileges
- Make sure no aufs is used (buggy and not supported)
- Docker bench for security audit tool
  - https://github.com/docker/docker-bench-security

Other considerations

- Always pin image version
  - Also, what about cached images? (--no-cache flag)
- Image and container sprawl
- Building context is important (what are we mounting?)
  - We can make use of .dockerignore file
  - Be careful not mounting docker.sock by mistake
- Reduce attack surface (coreOS, RancherOS, Red Hat Atomic, …)
- Avoid docker0 in production (ARP spoofing, MAC flooding)
- --pids-limit=100, --default-ulimit and --max-retries=5

Kubernetes

RBAC

- Actors
  - Users
  - ServiceAccounts
  - Groups (SA prefix, user cert organization field)
- Resources
  - Pod, deployment, …
- Role and ClusterRole
  - GET; WATCH; LIST; CREATE; UPDATE; PATCH; DELETE
- RoleBinding and ClusterRoleBinding

K8S Security Policy

- Pod/Container security context
  - runAsUser, dropping capabilities, …
- Adminission Controller
  - NodeRestriction
  - ValidatingAdminisionWebhooks (integrate with Anchore)
- Pod Security Policy
  - Which kind of Pod can we create?
- Kubernetes Network Policy

About secrets and certs

- Rotate, rotate, rotate
  - Probably we might want to use an external service
  - Audit them ;-)
- K8S only checks origin and expiration date
- How do I revoke access?
  - Create a new user and remove privileges to old one
  - Recreate the CA and re-issue again the certs
- What about ServiceAccounts tokens?
  - Delete and create again

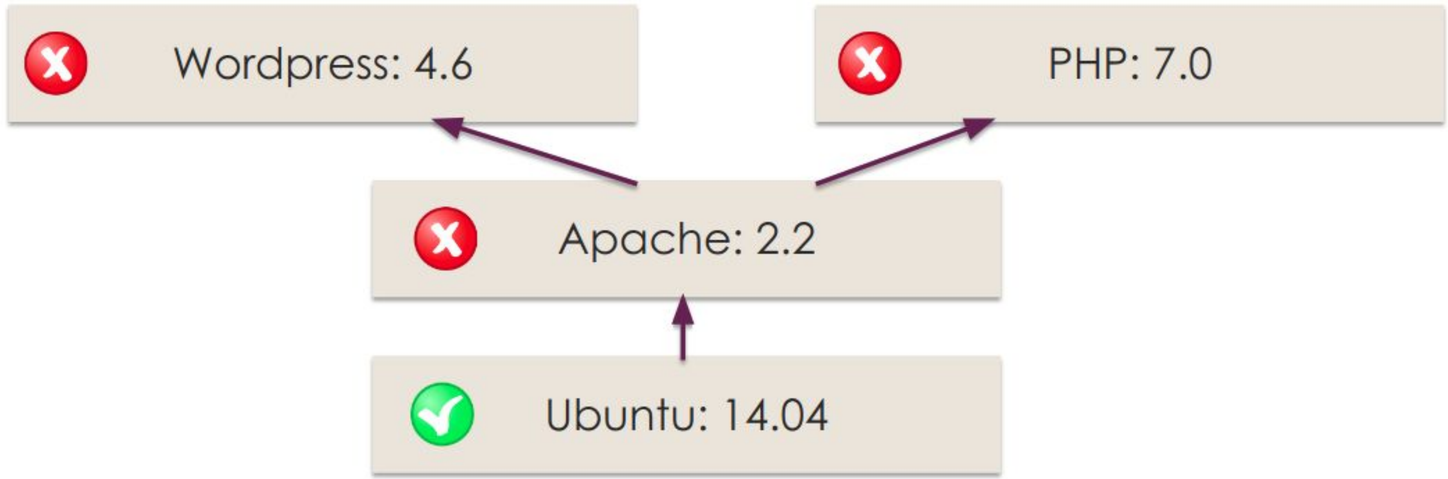
Let’s have some fun

https://github.com/figarocorso/security-workshop

Image Scanning

Why image scanning?

- Define at our company what is secure to be run in our arch
- Layered design is quite cool but



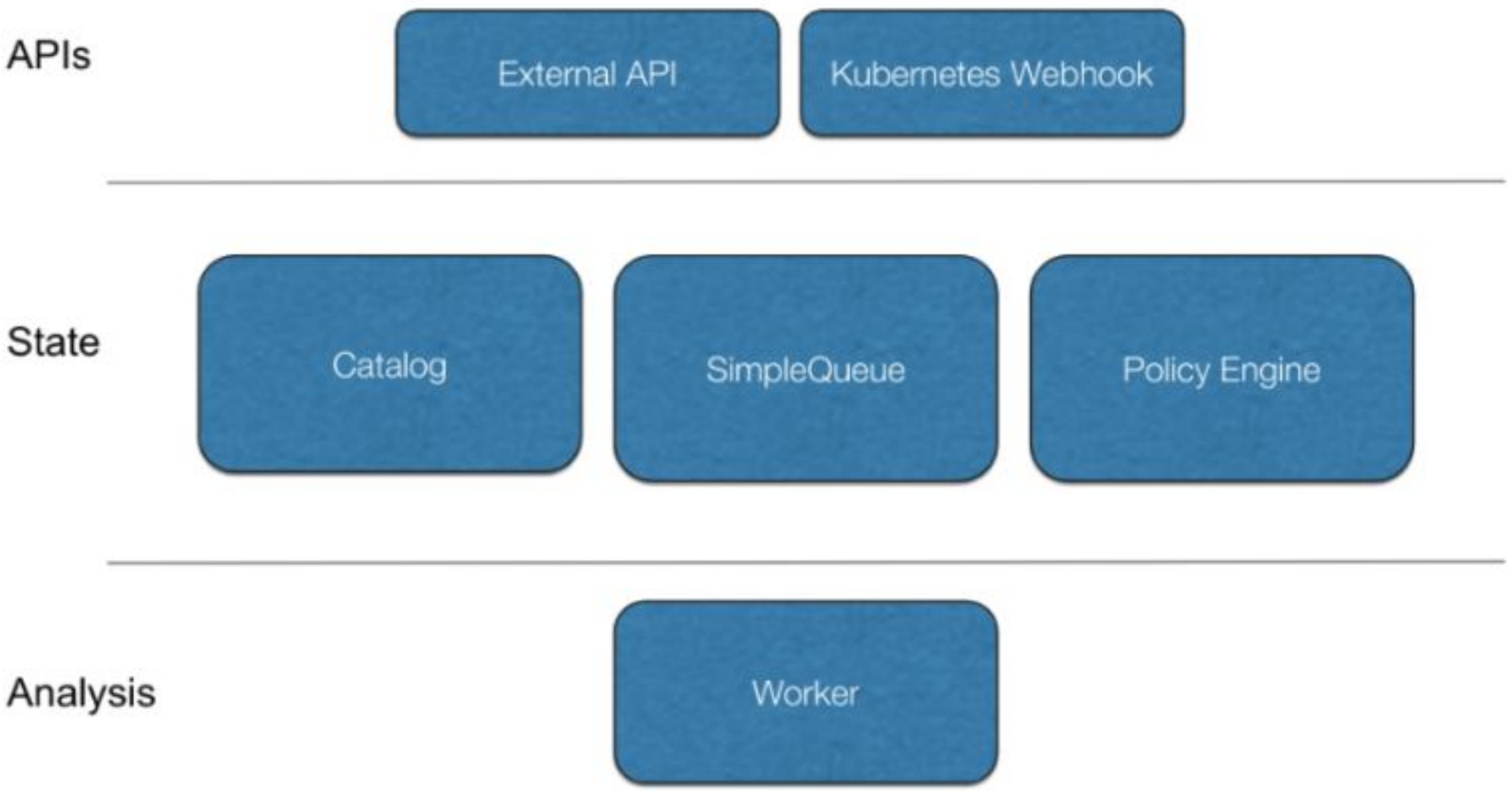
Alternatives

- Anchore Engine
  - Centralized service for inspection, analysis and user defined rules
- CoreOS/Clair
  - Static analysis in application containers
- Vuls.io
  - Linux vulnerability scanner (info from NVD and OVAL)
  - Some container image support, but not container specific tool
- OpenScap
  - Audit tool following the NIST-certtified SCAP
  - Again, not container specific

Image scanning

- Check pkgs, pypi, rubygem, binaries, OS files, ... against multiple vulnerabilities DB
- Whitelisted ones cached to speed-up process
- Analyze Dockerfile
  - Exposed ports
  - Privileged user (root)
  - Latest vs. pinned version
- User defined whiltelists, blacklist, policies

Anchore architecture



Let’s have some fun:

Deploy anchore

Let’s do some Jenkins magic

Runtime security and forensics

```
/usr/sbin/tcpdump {
  #include <abstractions/base>
  #include <abstractions/nameservice>
  #include <abstractions/user-tmp>

  capability net_raw,
  capability setuid,
  capability setgid,
  capability dac_override,
  network raw,
  network packet,

  # for -D
  capability sys_module,
  @{PROC}/bus/usb/ r,
  @{PROC}/bus/usb/** r,

  # for -F and -w
  audit deny @{HOME}/.* mrwkl,
  audit deny @{HOME}/.*/ rw,
  audit deny @{HOME}/.*/** mrwkl,
  audit deny @{HOME}/bin/ rw,
  audit deny @{HOME}/bin/** mrwkl,
  @{HOME}/ r,
  @{HOME}/** rw,

  /usr/sbin/tcpdump r,
```


Container security

- **Sandboxing: Seccomp**
  - **Seccomp: strict mode (read/write to already opened files)**
  - **Seccomp bpf: policies (letting each process a few privileges)**
- **Mandatory Access Control Systems: AppArmor**
  - **Permitted linux capabilities**
  - **Permitted network operations**
  - **Allowed/Disallowed files**
- **SELinux more complex, based in actor**

Container Forensics and Incident Response

Container forensic is difficult.
containers can be destroyed
rescheduled into a different node

Understand & Contain the impact of the SECURITY BREACH

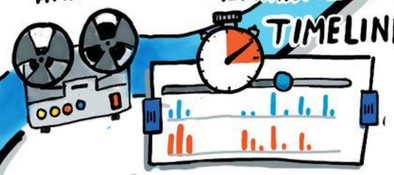
- ☑ WHAT HAPPENNED ?
- ☑ WHAT WAS the BREACH ?
- ☑ ANY DATA EXFILTRATED ?
- ☑ How DID THEY BREAK the SYSTEM?
- ☑ WHO DID IT ?

Like a TIME MACHINE

Cool feature!!

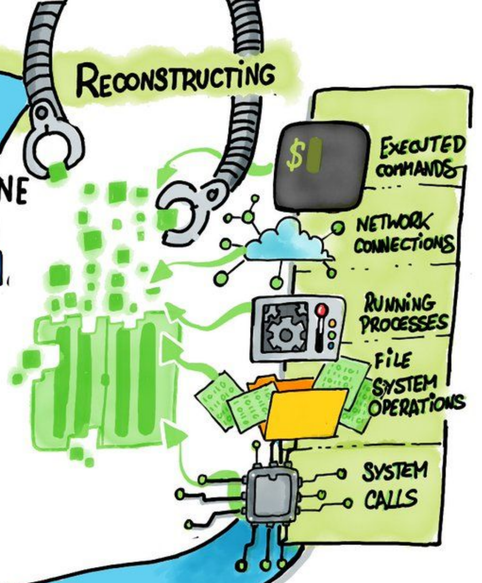
You need a way to perform forensics.
Even if your containers are long gone.

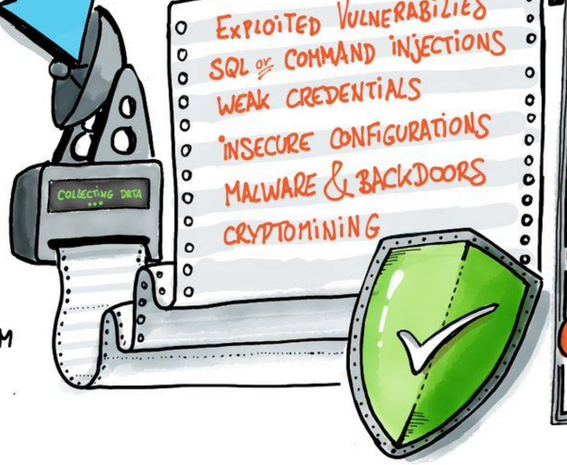
Recorded Pre & Post attack activity



Meaningful TIMELINE

Reconstructing





REPORT

- Exploited Vulnerabilies
- SQL or COMMAND injections
- WEAK CREDENTIALS
- INSECURE CONFIGURATIONS
- MALWARE & BACKDOORS
- CRYPTOMINING

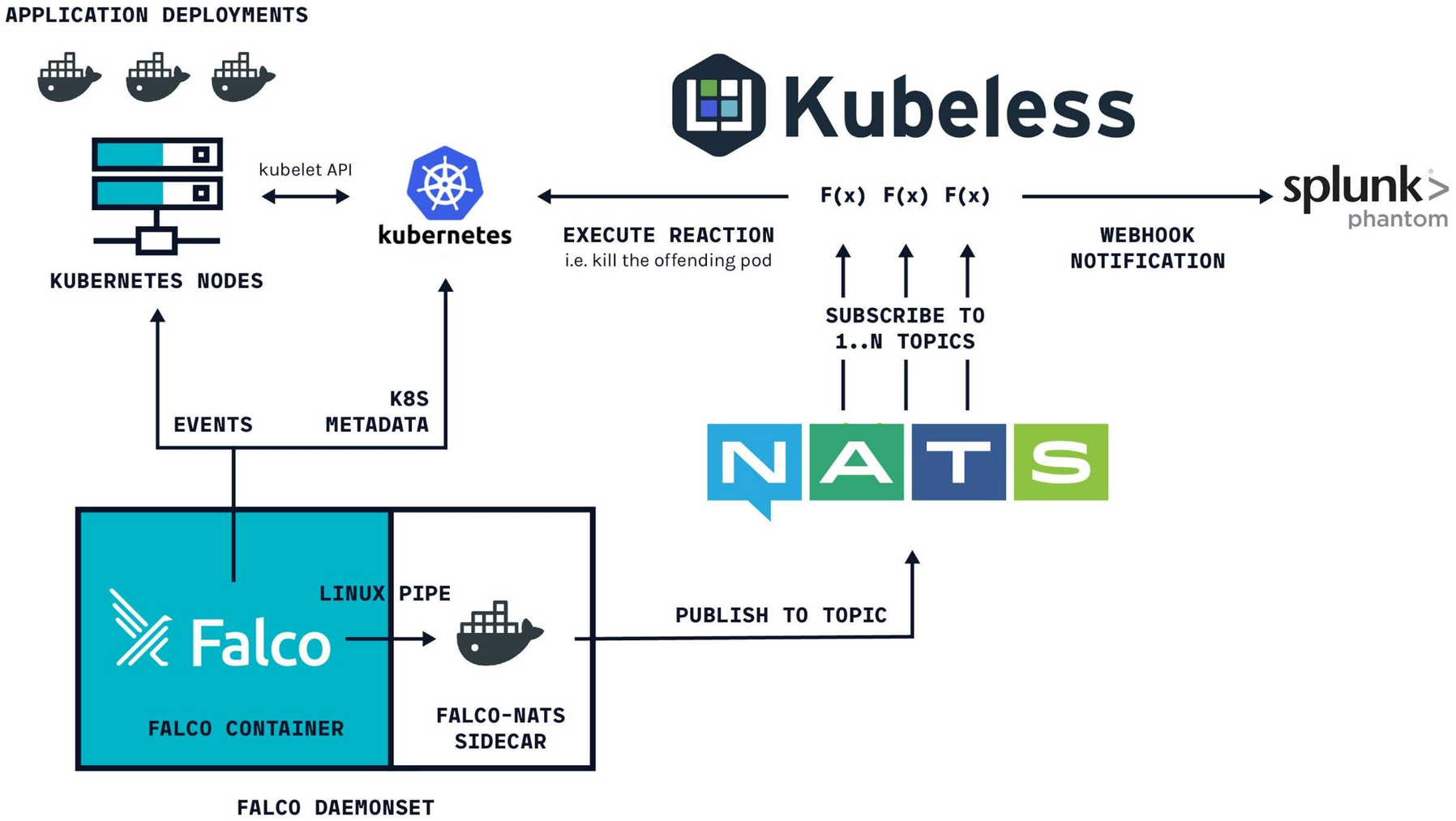
Sysdig

Behavioral monitoring

- **Falco!**
  - **Look at system calls as a event stream**
  - **As a userspace process: context!**
  - **Easy to write rules**
    - **So we only filter and analyze that syscall event stream**
  - **This rulz, but we might be biased :-)**

```
- rule: raw_network_socket
  desc: an attempt to open a raw network socket by an unexpected program
  condition: evt.type=socket and evt.dir=> and evt.arg.domain=AF_PACKET and not proc.name=tcpdump
  output: Raw network socket opened by unexpected program (user=%user.name command=%proc.cmdline domain=%evt.arg.domain)
  priority: WARNING
```

Reacting after an event



Let’s have some fun:
Launch the whole scenario

One step forward:

Let’s audit K8S events

Thank you!!

- Miguel Julián
  - @miky_kun
  - miguel.julian@sysdig.com
- Julio García
  - @papajulio
  - julio.garcia@sysdig.com