Automatically trigger captures via tcpdump when a suspicious event occurs in your Kubernetes cluster

Thomas Labarussias

Let me introduce myself

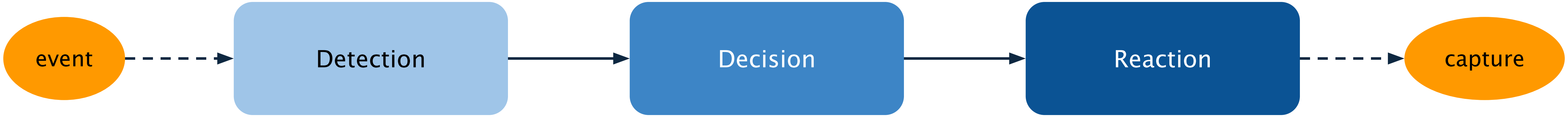
Thomas Labarussias

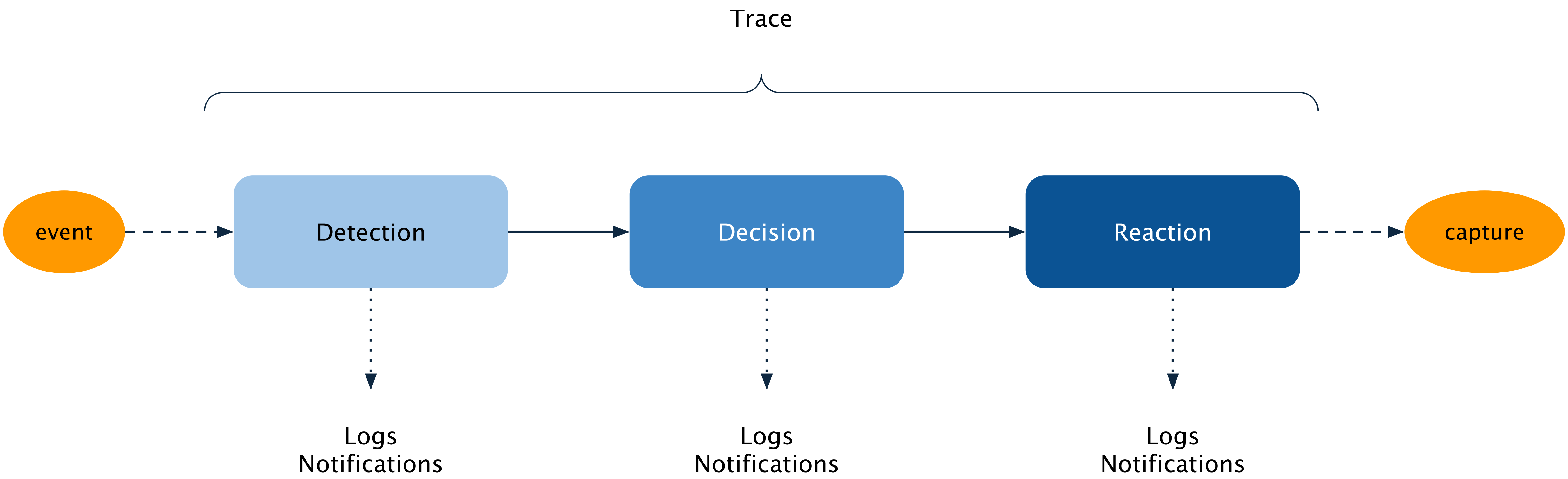
Senior Developer Advocate at **Sysdig**
SRE for 8y⁺ + FinOps 3y
CNCF Ambassador

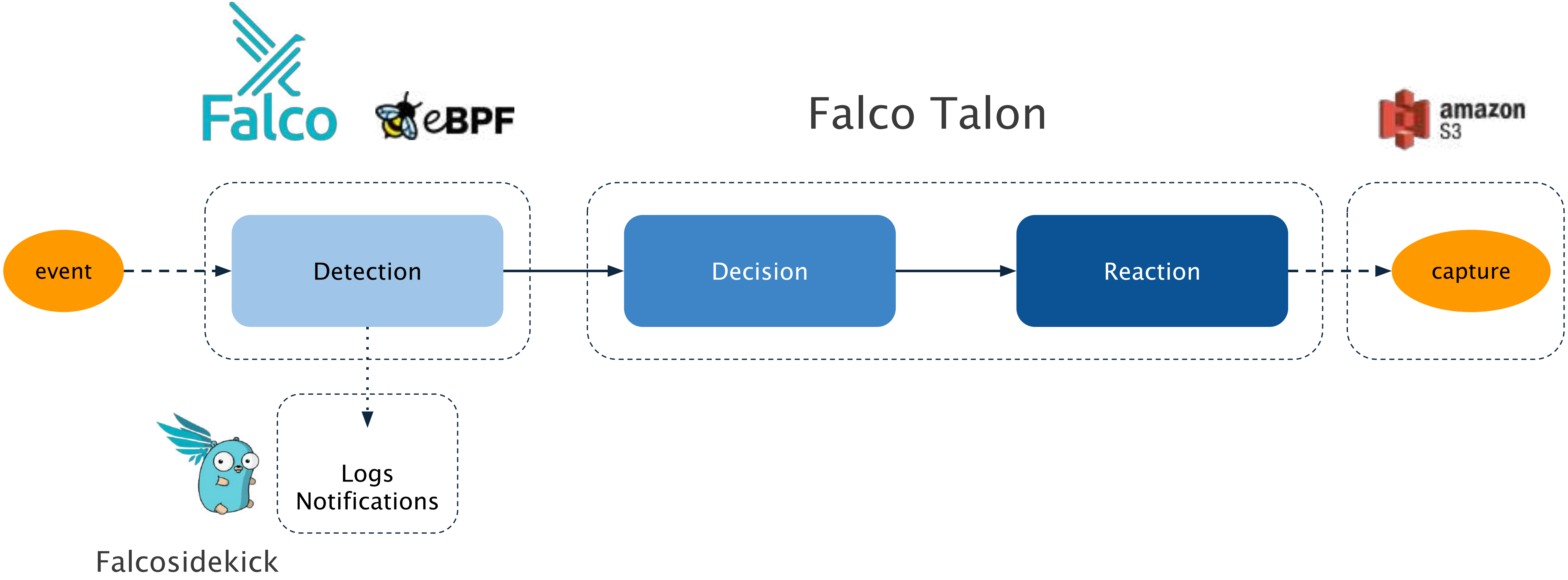
Falco contributor
Creator of Falcosidekick and Talon

github.com/Issif
@TLabarussias
untappd.com/user/Issif

Automatically trigger captures via tcpdump when a suspicious event occurs in your Kubernetes cluster


event
Detection
Decision
Reaction
capture


Trace
event
Detection
Decision
Reaction
capture
Logs
Notifications
Logs
Notifications
Logs
Notifications



Falco

eBPF

Falco Talon

amazon S3

event

Detection

Decision

Reaction

capture

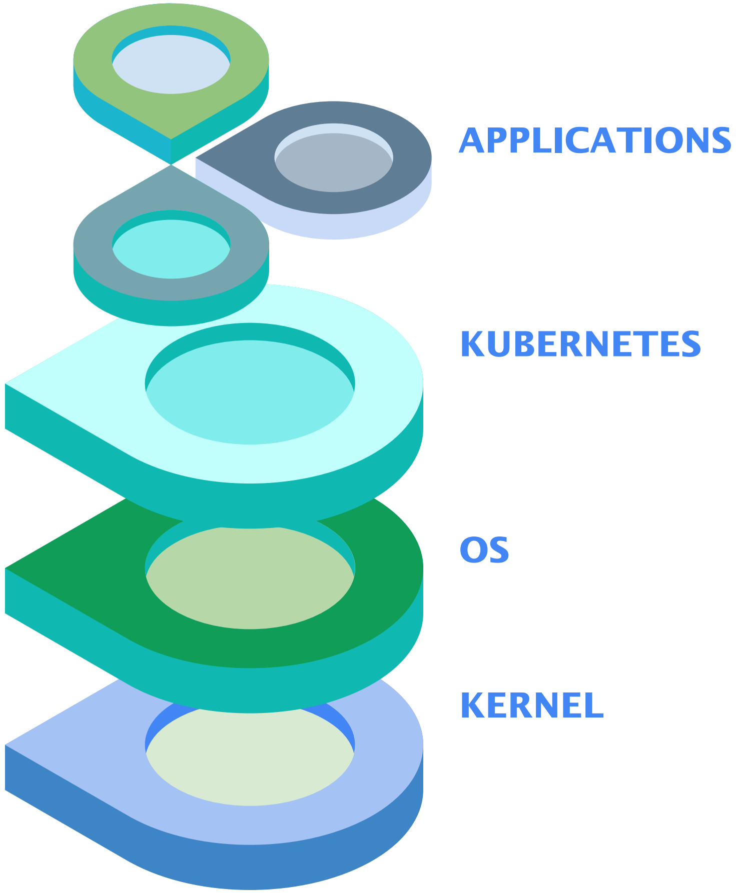
Logs
Notifications

Falcosidekick

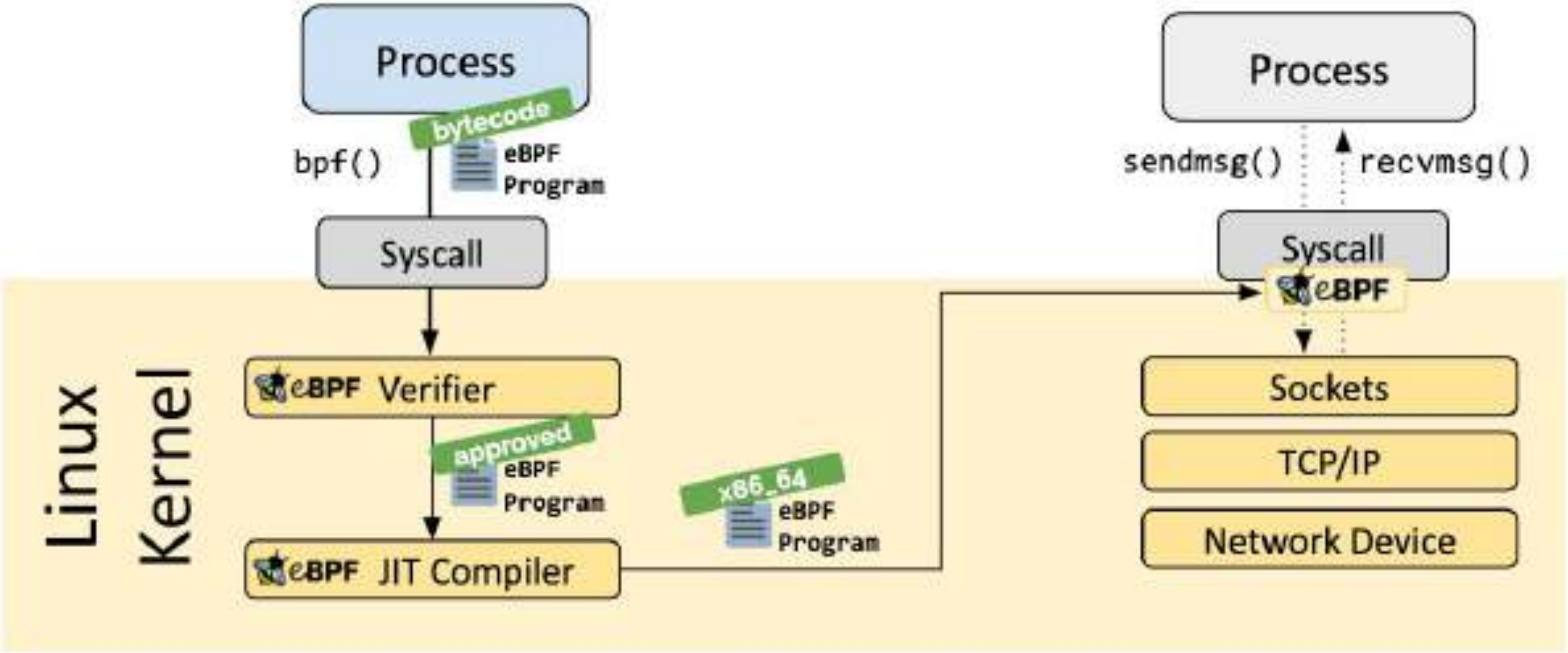
Syscalls

System Calls are the way for programs to ask the Kernel for access to resources.

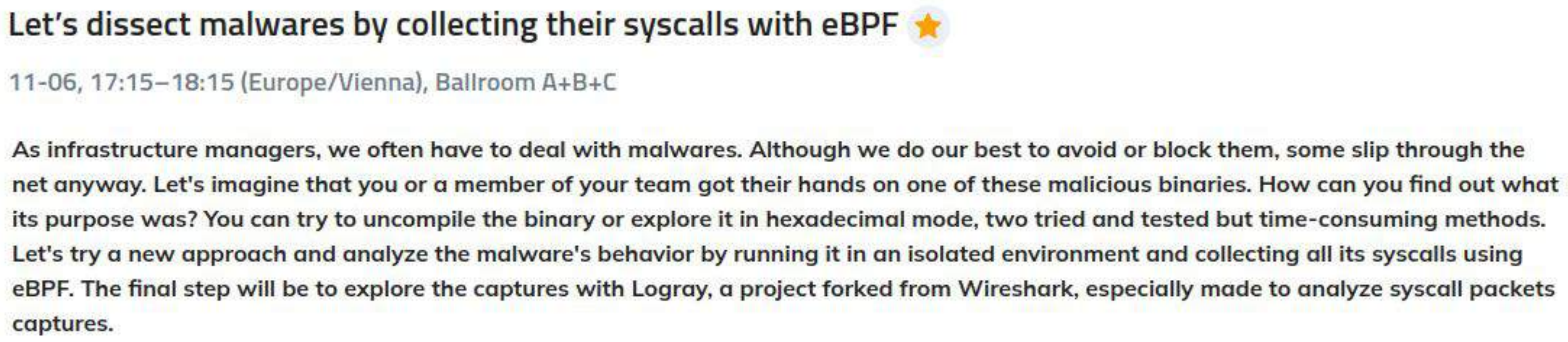
- process
- network
- IO files
- and more…



- **Linux Kernel feature** that lets you run programs in the Kernel **without modifying its code** or loading a module
- Accesses **kernel** activity without any risk for stability or security
- Used for **security**, **monitoring** or **diagnostics**



https://ebpf.io



Let's dissect malwares by collecting their syscalls with eBPF ★

11-06, 17:15–18:15 (Europe/Vienna), Ballroom A+B+C

As infrastructure managers, we often have to deal with malwares. Although we do our best to avoid or block them, some slip through the net anyway. Let's imagine that you or a member of your team got their hands on one of these malicious binaries. How can you find out what its purpose was? You can try to uncompile the binary or explore it in hexadecimal mode, two tried and tested but time-consuming methods. Let's try a new approach and analyze the malware's behavior by running it in an isolated environment and collecting all its syscalls using eBPF. The final step will be to explore the captures with Logray, a project forked from Wireshark, especially made to analyze syscall packets captures.

Detection: Falco

Falco

CLOUD NATIVE
COMPUTING FOUNDATION

A graduated CNCF project

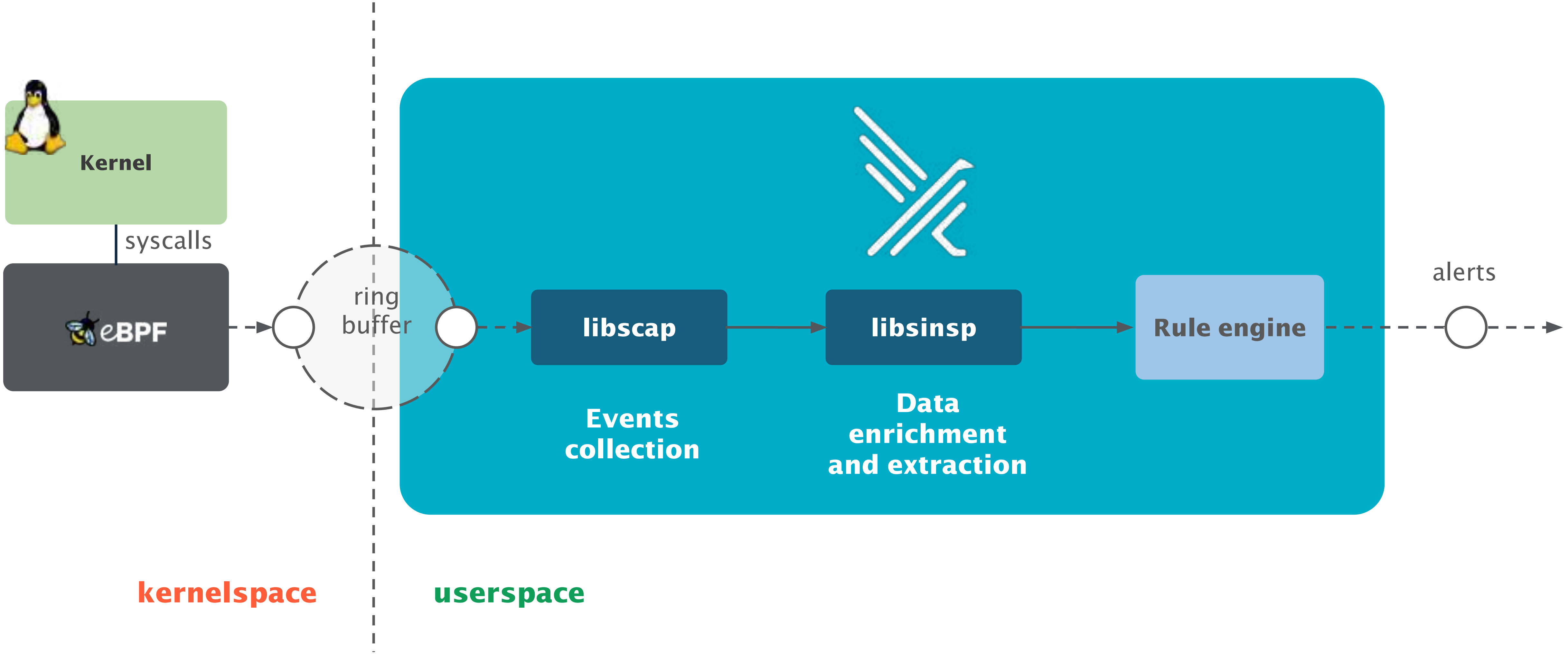
Falco, a cloud-native project to **secure running applications** by **detecting threats** in **Kubernetes** clusters, **Cloud** environments, **Linux** hosts and **more**.

Powered by **eBPF**
Plugins for extra sources

⭐ 7k+
🐳 120M+ pulls

https://falco.org

Falco's architecture



https://falco.org

Falco Rules

```
- rule: Netcat Remote Code Execution in Container
  desc: >
    Netcat Program runs inside container that allows remote code execution and may be
utilized as a part of a variety of reverse shell payload
  condition: >
    spawned_process and container and
    ((proc.name = "nc" and (proc.cmdline contains " -e" or proc.cmdline contains " -c")) or
    (proc.name = "ncat" and
        (proc.args contains "--sh-exec" or
         proc.args contains "--exec" or proc.args contains "-e " or
         proc.args contains "-c " or proc.args contains "--lua-exec")))
  output: >
    Netcat runs inside container that allows remote code execution (evt_type=%evt.type
user=%user.name user_uid=%user.uid user_loginuid=%user.loginuid process=%proc.name
proc_exepath=%proc.exepath parent=%proc.pname command=%proc.cmdline terminal=%proc.tty
exe_flags=%evt.arg.flags %container.info)
  priority: WARNING
  tags: [maturity_stable, container, network, process, mitre_execution, T1059]
```

Falco Rules

```
- rule: Netcat Remote Code Execution in Container
  desc: >
    Netcat Program runs inside container that allows remote code execution and may be
utilized as a part of a variety of reverse shell payload
  condition: >
    spawned_process and container and
    ((proc.name = "nc" and (proc.cmdline contai                                        or
    (proc.name = "ncat" and
        (proc.args contains "--sh-exec" or
         proc.args contains "--exec" or proc.arg
         proc.args contains "-c " or proc.args c
  output: >
    Netcat runs inside container that allows rem
user=%user.name user_uid=%user.uid user_loginuic
proc_exepath=%proc.exepath parent=%proc.pname command=%proc.cmdline terminal=%proc.tty
exe_flags=%evt.arg.flags %container.info)
  priority: WARNING
  tags: [maturity_stable, container, network, process, mitre_execution, T1059]
```

```
- macro: spawned_process
  condition: >
        evt.type in (execve, execveat)
        and evt.dir=<

- macro: container
  condition: (container.id != host)
```

Falco Default Rules

- Privilege escalation
- R/W to sensitive directories
- Executing shell
- Execute SSH/Network binaries
- Mutating binaries
- Creating symlinks
- Data exfiltration
- ...

80+ system rules

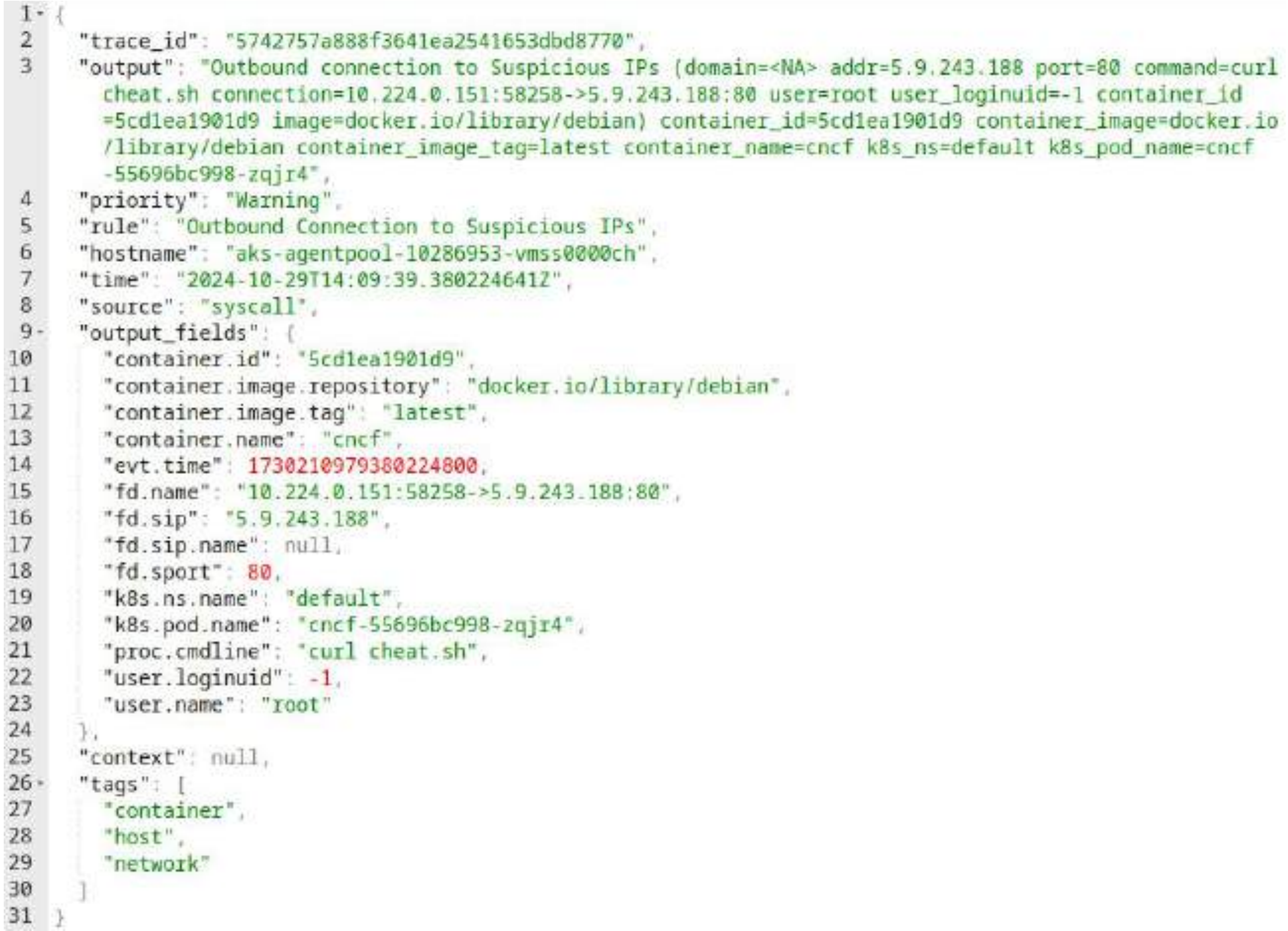
All customizable:

```
- list: shell_binaries
  items: [fish]
  override:
    items: append

- rule: Terminal shell in container
  condition: >
        and not k8s.ns.name=kube-system
  override:
    condition: append
```

https://github.com/falcosecurity/rules

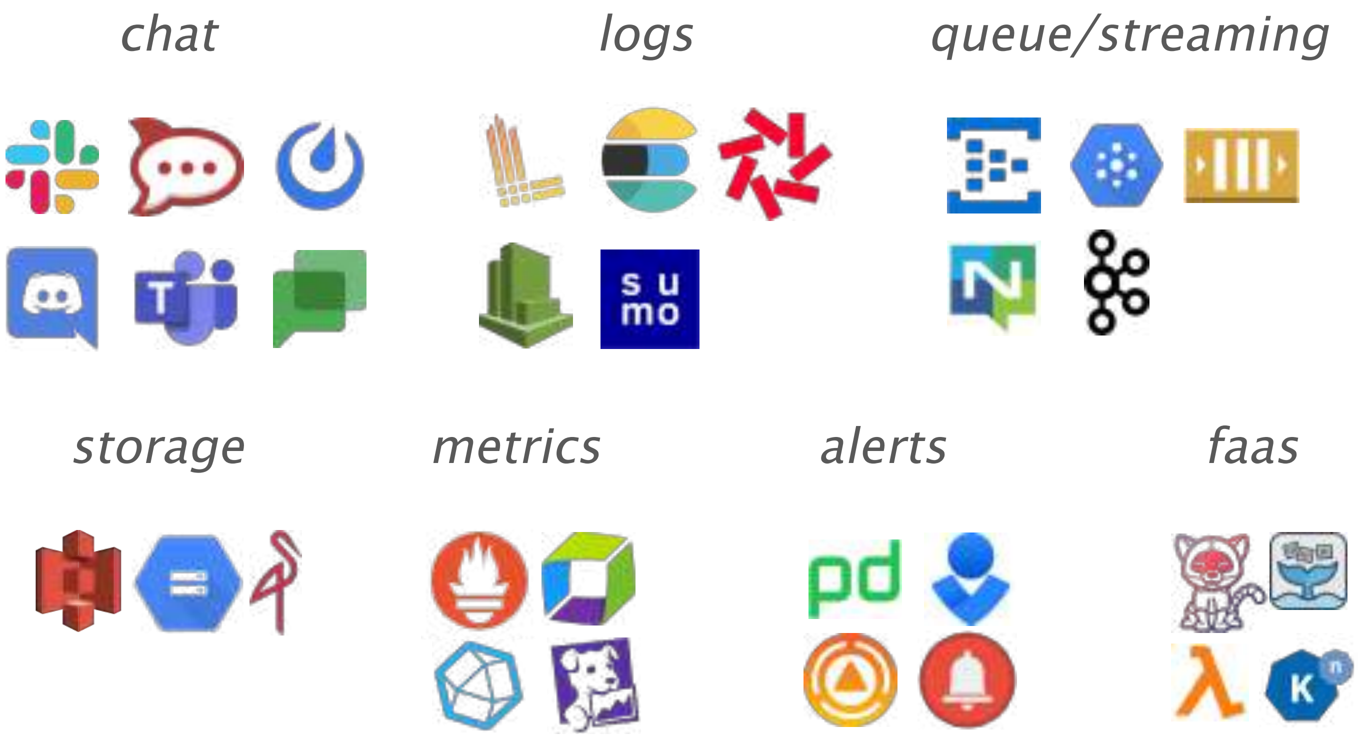
Falco Alerts



```
{
  "trace_id": "5742757a888f3641ea2541653dbd8770",
  "output": "Outbound connection to Suspicious IPs (domain=<NA> addr=5.9.243.188 port=80 command=curl cheat.sh connection=10.224.0.151:58258->5.9.243.188:80 user=root user_loginuid=-1 container_id=5cd1ea1901d9 image=docker.io/library/debian) container_id=5cd1ea1901d9 container_image=docker.io/library/debian container_image_tag=latest container_name=cncf k8s_ns=default k8s_pod_name=cncf-55696bc998-zqjr4",
  "priority": "Warning",
  "rule": "Outbound Connection to Suspicious IPs",
  "hostname": "aks-agentpool-10286953-vmss0000ch",
  "time": "2024-10-29T14:09:39.380224641Z",
  "source": "syscall",
  "output_fields": {
    "container.id": "5cd1ea1901d9",
    "container.image.repository": "docker.io/library/debian",
    "container.image.tag": "latest",
    "container.name": "cncf",
    "evt.time": 1730210979380224800,
    "fd.name": "10.224.0.151:58258->5.9.243.188:80",
    "fd.sip": "5.9.243.188",
    "fd.sip.name": null,
    "fd.sport": 80,
    "k8s.ns.name": "default",
    "k8s.pod.name": "cncf-55696bc998-zqjr4",
    "proc.cmdline": "curl cheat.sh",
    "user.loginuid": -1,
    "user.name": "root"
  },
  "context": null,
  "tags": [
    "container",
    "host",
    "network"
  ]
}
```

Notifications: Falcosidekick

70+ integrations



chat

logs

queue/streaming

storage

metrics

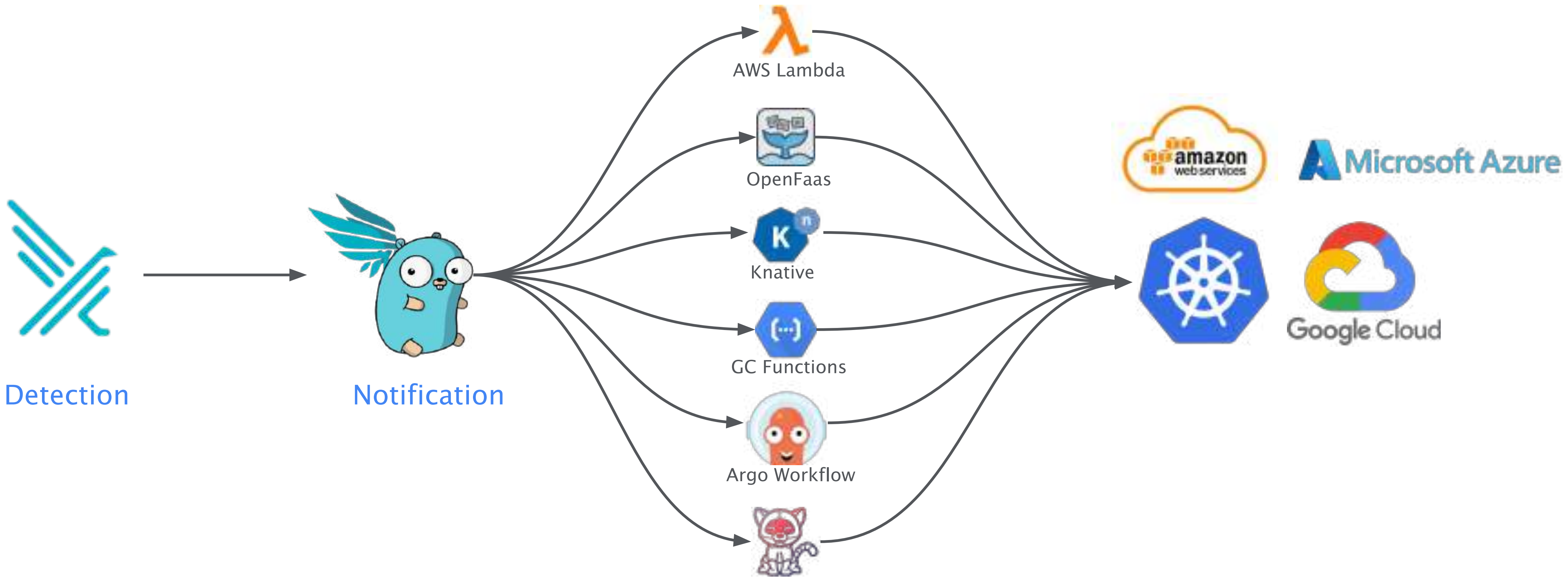
alerts

faas

https://github.com/falcosecurity/falcosidekick

Reaction: How to?

SharkFest'24 EUROPE
Vienna, Austria ▪ #sf24eu



Detection

Notification

Reaction

Developing your own "response engine"

Benefits

- Total flexibility
- Total control over actions
- Not dependent on a third party (for fixes and updates)
- Allows you to use services and procedures already in place

Developing your own “response engine”

Benefits

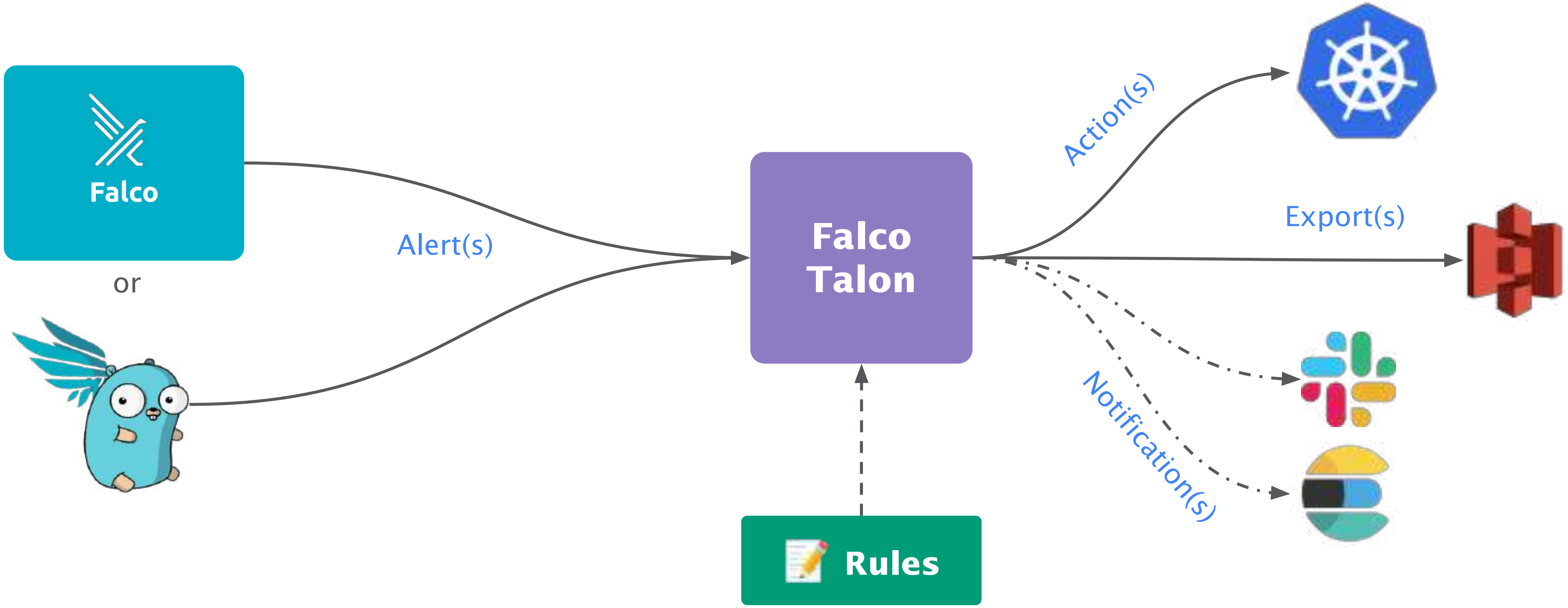
- Total flexibility
- Total control over actions
- Not dependent on a third party (for fixes and updates)
- Allows you to use services and procedures already in place

Drawbacks

- Need to develop actions, manage errors, authentications, logs, notifications, etc.
- SDK complexity management for K8S, Clouds, ...
- May require installation/management of new services
- Latency if external
- Complexity of chaining multiple actions

Reaction: Falco Talon

A no-code Response Engine natively incorporating Falco alerts, enabling actions to be triggered according to rules



https://github.com/falcosecurity/falco-talon

Reaction: Falco Talon

- Zero code
  - Yaml rules files
- Available Actions (more are coming):
  - kubernetes:terminate
  - kubernetes:label
  - kubernetes:networkpolicy
  - kubernetes:exec
  - kubernetes:script
  - kubernetes:log
  - kubernetes:delete
  - kubernetes:cordon
  - kubernetes:drain
  - **kubernetes:tcpdump**
  - kubernetes:download
  - calico:networkpolicy
  - cilium:networkpolicy
  - aws:lambda

- The actions are triggered by conditions based on:
  - priority
  - tags
  - source
  - Falco rule name
  - output fields
- Sequential actions
- Export of artifacts (AWS S3, Minio)
- Deduplication of the Falco alerts
- Out of the box notifiers

  (Slack, Email, Webhook, Loki, Elasticsearch, K8S Events)
- Structured logs (with a traceID to follow the steps)
- OTLP Traces + Prometheus metrics

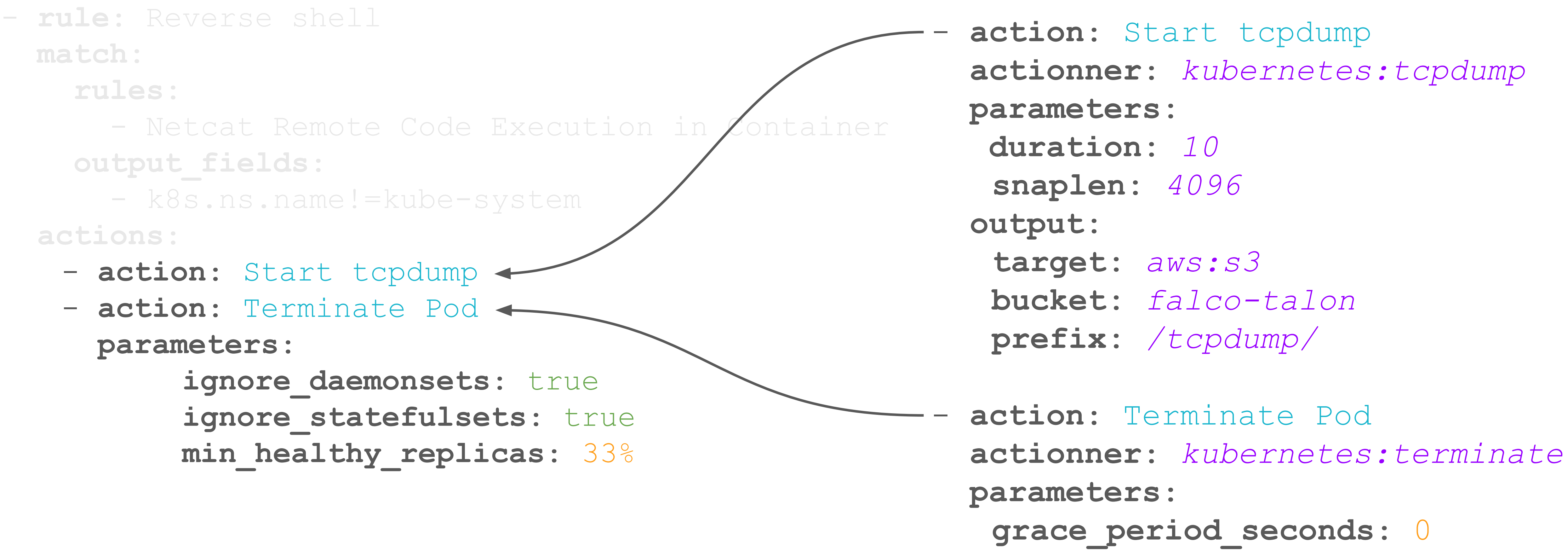
Falco Talon rules

```
- rule: Reverse shell
  match:
    rules:
      - Netcat Remote Code Execution in Container
    output_fields:
      - k8s.ns.name!=kube-system
  actions:
   - action: Start tcpdump
   - action: Terminate Pod
     parameters:
          ignore_daemonsets: true
          ignore_statefulsets: true
          min_healthy_replicas: 33%
```

```
- action: Start tcpdump
  actionner: kubernetes:tcpdump
  parameters:
   duration: 10
   snaplen: 4096
  output:
   target: aws:s3
   bucket: falco-talon
   prefix: /logs/

- action: Terminate Pod
  actionner: kubernetes:terminate
  parameters:
   grace_period_seconds: 0
```

Falco Talon rules

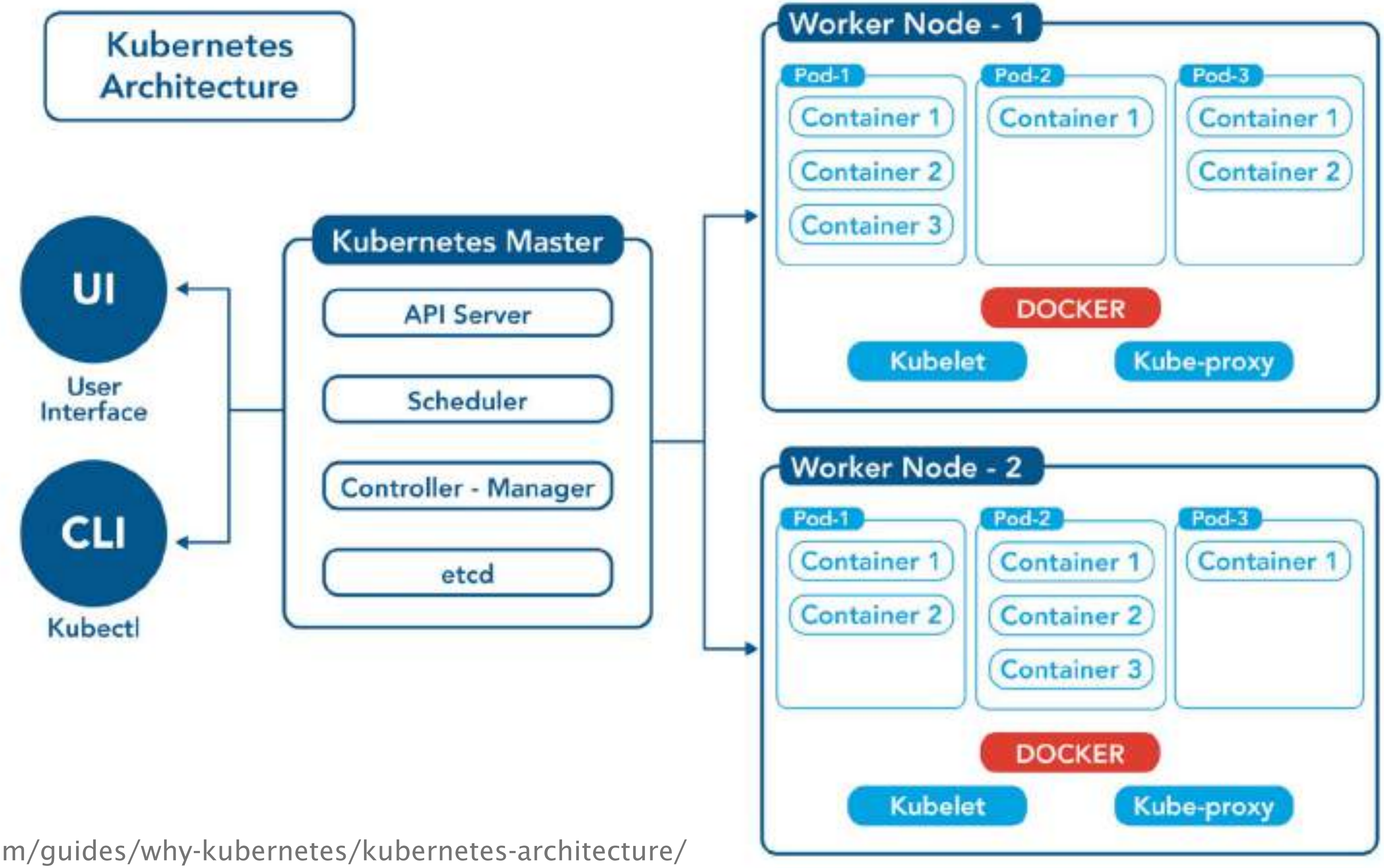


```
- rule: Reverse shell
  match:
    rules:
      - Netcat Remote Code Execution in Container
    output_fields:
      - k8s.ns.name!=kube-system
  actions:
   - action: Start tcpdump
   - action: Terminate Pod
     parameters:
         ignore_daemonsets: true
         ignore_statefulsets: true
         min_healthy_replicas: 33%
```

```
- action: Start tcpdump
  actionner: kubernetes:tcpdump
  parameters:
   duration: 10
    snaplen: 4096
  output:
    target: aws:s3
    bucket: falco-talon
    prefix: /tcpdump/
```

```
- action: Terminate Pod
  actionner: kubernetes:terminate
  parameters:
    grace_period_seconds: 0
```

Kubernetes 101



https://www.opsramp.com/guides/why-kubernetes/kubernetes-architecture/

How to run a tcpdump in Kubernetes?

Different methods:

SSH

- − Doesn't work with managed clusters
- − Need to find the veth attached to the container
- + Full flexibility

How to run a tcpdump in Kubernetes?

Different methods:

SSH

− Doesn’t work with managed clusters

− Need to find the veth attached to the container

+ Full flexibility

Exec inside a container of the pod

− Need a root user

− Need a shell env + tcpdump binary

+ Immediate action (if tcpdump is present)

How to run a tcpdump in Kubernetes?

Different methods:

SSH

- − Doesn’t work with managed clusters
- − Need to find the veth attached to the container
- + Full flexibility

Start a new pod on the same node

- − Need to find the veth and mount it
- − Possible latency because of the image pull
- + Full flexibility

Exec inside a container of the pod

- − Need a root user
- − Need a shell env + tcpdump binary
- + Immediate action (if tcpdump is present)

How to run a tcpdump in Kubernetes?

Different methods:

SSH

− Doesn’t work with managed clusters

− Need to find the veth attached to the container

+ Full flexibility

Start a new pod on the same node

− Need to find the veth and mount it

− Possible latency because of the image pull

+ Full flexibility

Exec inside a container of the pod

− Need a root user

− Need a shell env + tcpdump binary

+ Immediate action (if tcpdump is present)

Run an ephemeral container

− Need a root user

− Possible latency because of the image pull

+ Original veth is shared

How to run a tcpdump in Kubernetes?

Different methods:

SSH

- − Doesn't work with managed clusters
- − Need to find the veth attached to the container
- + Full flexibility

Exec inside a container of the pod

- − Need a root user
- − Need a shell env + tcpdump binary
- + Immediate action (if tcpdump is present)

Start a new pod on the same node

- − Need to find the veth and mount it
- − Possible latency because of the image pull
- + Full flexibility

Run an ephemeral container

- − Need a root user
- − Possible latency because of the image pull
- + Original veth is shared

Ephemeral container

In Kubernetes, **Pods** are **immutable**

Ephemeral containers

“a special type of container that runs temporarily in an existing Pod to accomplish user-initiated actions”

Mostly used for **troubleshooting**, they allow to run binaries not present in the original containers of the pod

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    app: cncf
  name: cncf-55696bc998-8hqmm
  namespace: default
spec:
  containers:
    - command:
        - sleep
        - infinity
      image: debian
      imagePullPolicy: Always
      name: cncf
  nodeName: aks-agentpool-10286953-vmss0000c9
```

https://kubernetes.io/docs/concepts/workloads/pods/ephemeral-containers/

Ephemeral container

In Kubernetes, **Pods** are **immutable**

Ephemeral containers

“a special type of container that runs temporarily in an existing Pod to accomplish user-initiated actions”

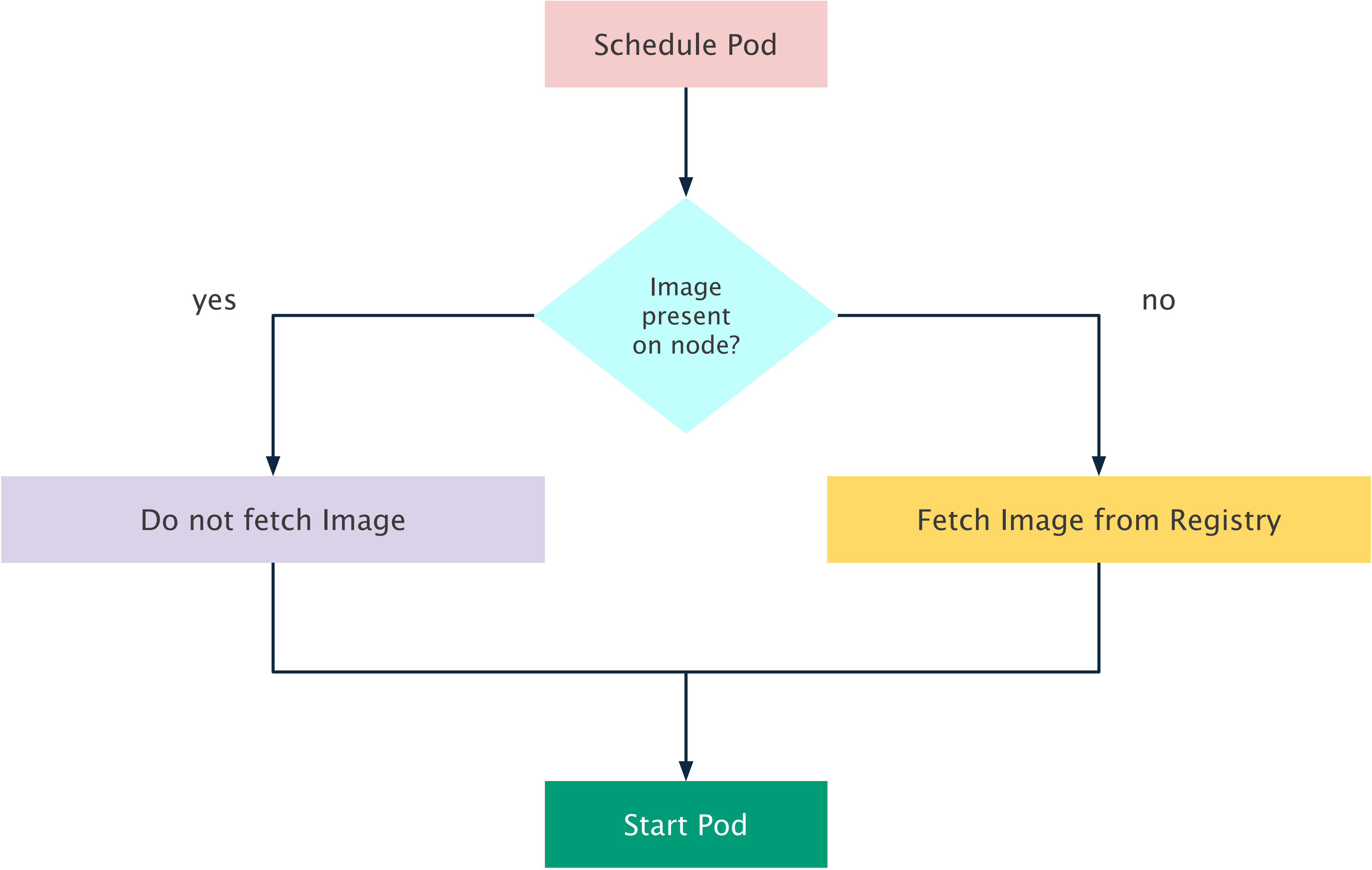
Mostly used for **troubleshooting**, they allow to run binaries not present in the original containers of the pod

The mechanism behind kubectl debug

https://kubernetes.io/docs/concepts/workloads/pods/ephemeral-containers/

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    app: cncf
  name: cncf-55696bc998-8hqmm
  namespace: default
spec:
  containers:
    - command:
        - sleep
        - infinity
      image: debian
      imagePullPolicy: Always
      name: cncf
  ephemeralContainers:
    - command:
        - sleep
        - "10"
      image: busybox
      imagePullPolicy: Always
      name: debugger-wv2jg
      resources: {}
      securityContext:
        capabilities:
          add:
            - SYS_PTRACE
      stdin: true
      tty: true
  nodeName: aks-agentpool-10286953-vmss0000c9
```

Image pull latency issue



Schedule Pod

Image present on node?

yes

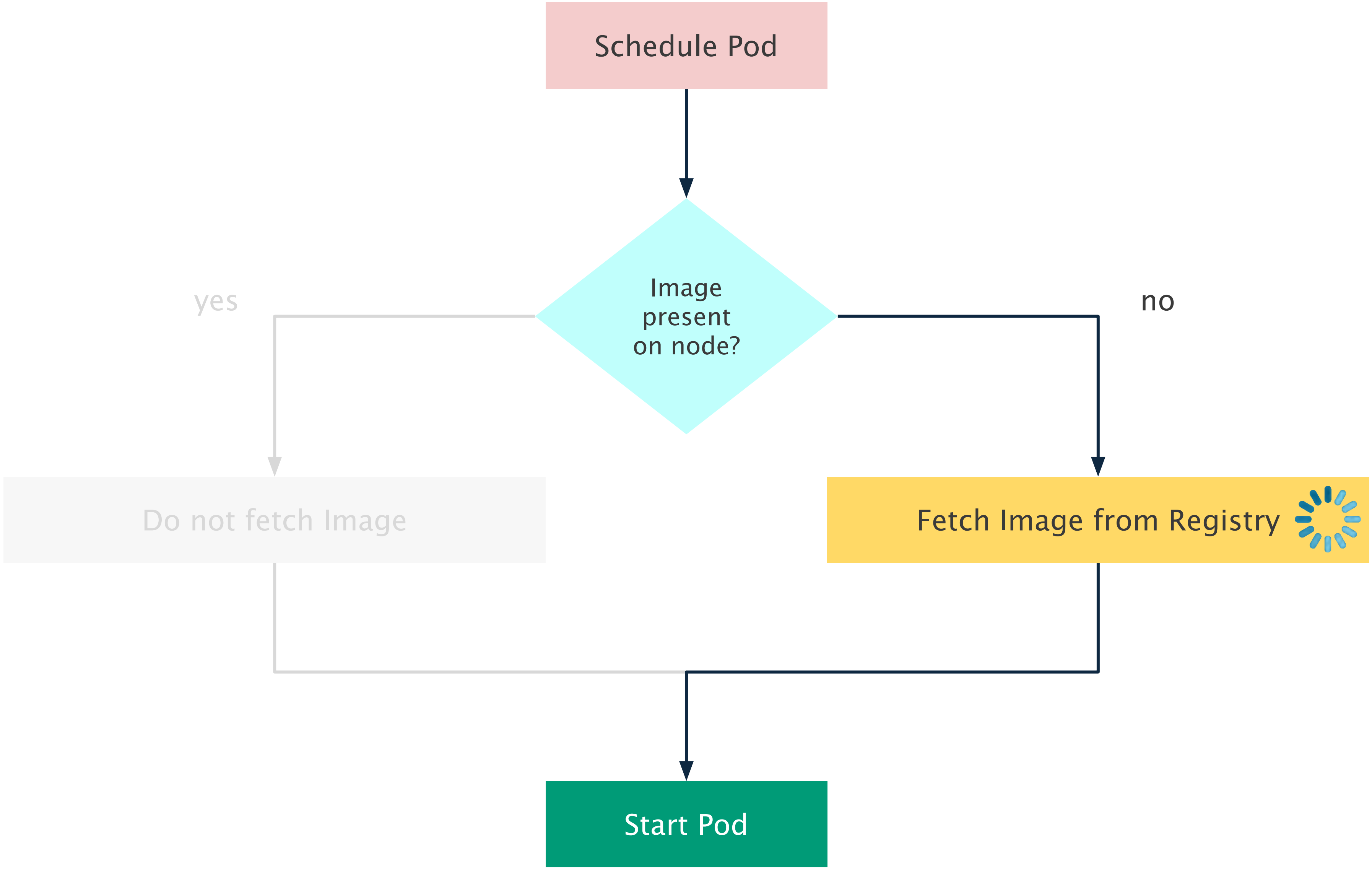
no

Do not fetch Image

Fetch Image from Registry

Start Pod

Image pull latency issue



Schedule Pod

Image present on node?

yes

no

Do not fetch Image

Fetch Image from Registry

Start Pod

Image pull latency issue

```
REPOSITORY          TAG          IMAGE ID       CREATED         SIZE
ubuntu_tcpdump      latest       6d2dcfe47029   7 months ago    131MB
```

Image pull latency issue

```
REPOSITORY          TAG          IMAGE ID        CREATED          SIZE
ubuntu_tcpdump      latest       6d2dcfe47029    7 months ago     131MB
```

Slim toolkit

```
slim build --target dockersec/tcpdump:latest --tag issif/tcpdump:latest
--http-probe=false --exec "sh -c \"sleep 1\"; timeout 10s tcpdump -i any -W 1 -G 5 -w
/tmp/cap.cap; ls -al /tmp/cap.cap | tee /tmp/cap.txt /dev/null; cat /tmp/cap.txt"
```

Image pull latency issue

```
REPOSITORY          TAG          IMAGE ID        CREATED          SIZE
ubuntu_tcpdump      latest       6d2dcfe47029    7 months ago     131MB
```

Slim toolkit

```
slim build --target dockersec/tcpdump:latest --tag issif/tcpdump:latest
--http-probe=false --exec "sh -c \"sleep 1\"; timeout 10s tcpdump -i any -W 1 -G 5 -w
/tmp/cap.cap; ls -al /tmp/cap.cap | tee /tmp/cap.txt /dev/null; cat /tmp/cap.txt"
```

```
REPOSITORY          TAG          IMAGE ID        CREATED          SIZE
ubuntu_tcpdump      small        cc7bf4810fc6    2 week ago       13.6MB
```

Demo

Feedback