Using Sysdig Falco & EFK Logging and Monitoring

Falco is the open source standard for runtime security for hosts, containers, Kubernetes and the cloud. Get real-time visibility into unexpected behaviors, config changes, intrusions, and data theft.

What makes Falco Different?

- Cloud Native
    - Falco detects threats across containers, Kubernetes, hosts and cloud services.

- Real Time Protection
    - Falco provides streaming detection of unexpected behavior, configuration changes, and attacks.
- Open source
    - A multi-vendor and broadly supported standard that you can rely on.

What does Falco do?

- Falco uses system calls to secure and monitor a system, by:
    - Parsing the Linux system calls from the kernel at runtime.
    - Asserting the stream against a powerful rules engine.
    - Alerting when a rule is violated.

What does Falco check for?

- Falco ships with a default set of rules that check the kernel for unusual behavior such as:
    - Privilege escalation using privileged containers
    - Namespace changes using tools like setns
    - Read/Writes to well-known directories such as /etc, /usr/bin, /usr/sbin, etc
    - Creating symlinks
    - Ownership and Mode changes
    - Unexpected network connections or socket mutations
    - Spawned processes using execve
    - Executing shell binaries such as sh, bash, csh, zsh, etc
    - Executing SSH binaries such as ssh, scp, sftp, etc
    - Mutating Linux coreutils executables
    - Mutating login binaries
    - Mutating shadowutil or passwd executables such as shadowconfig, pwck, chpasswd, getpasswd, change, useradd, etc, and others.

What are Falco rules?

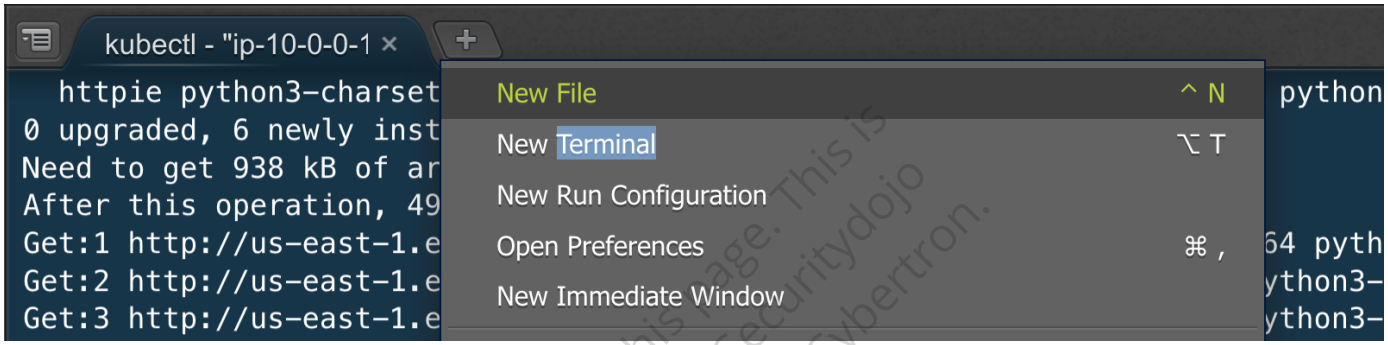
- Rules are the items that Falco asserts against. They are defined in the Falco

configuration file, and represent the events you can check on the system. For more information about writing, managing, and deploying rules, see Falco Rules.

Hands on Lab Falco: Runtime security monitoring & detection

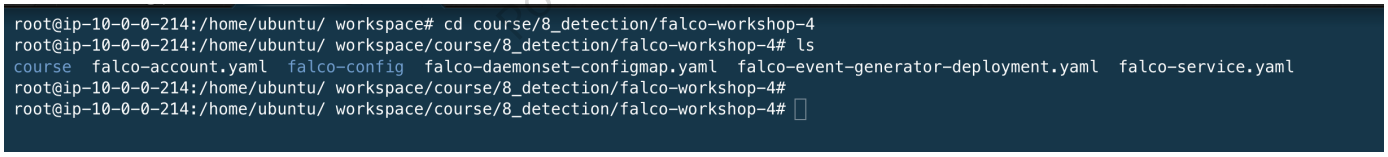
Open New Terminal (Optional)

- Click on `+` icon, then select `new terminal` to open new terminal.



- Keep current working directory as `workspace/`

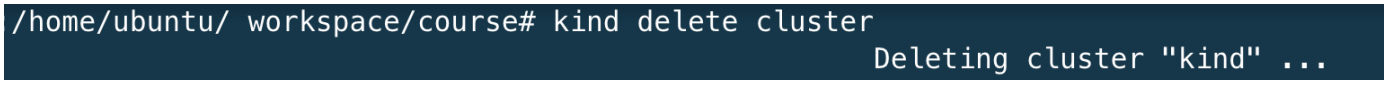
```
cd course/8_detection/falco-workshop-4
ls
```



- Delete the priviously created kind cluster.

For this setup, cluster via kubeadm would be installed.

```
kind delete cluster
```



- Install kubeadm and kubelet binary.

```
apt -y install kubeadm=1.22.0-00 kubectl kubelet=1.22.0-00
```

This version is used because lab was tested and created using mentioned version.

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# apt -y install kubeadm=1.22.0-00 kubectl kubelet=1.22.0-00
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libintl-perl libintl-xs-perl libmodule-find-perl libmodule-scandeps-perl libproc-processtable-perl libsort-naturally-perl libterm-readkey-perl
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
The following NEW packages will be installed:
  conntrack cri-tools ebtables kubeadm kubectl kubelet kubernetes-cni socat
0 upgraded, 8 newly installed, 0 to remove and 85 not upgraded.
Need to get 87.9 MB of archives.
```

- Hold the kubelet, kubeadm & kubectl version.

```
apt-mark hold kubelet kubeadm kubectl
```



```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# apt-mark hold kubelet kubeadm kubectl
                                                                                                                kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Disable swap memory, the comment out swap entries in the filesystem table to prevent them from being mounted on boot.
- Finally, Load the overlay kernel module, which supports overlay filesystems in Docker and container runtimes.

```
swapoff -a
sed -i.bak -r 's/(.+ swap .+)/#\1/' /etc/fstab
modprobe overlay
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# swapoff -a
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# sed -i.bak -r 's/(.+ swap .+)/#\1/' /etc/fstab
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# modprobe overlay
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Writes a Docker configuration to use overlay2 as storage to "/etc/docker/daemon.json".

```
tee /etc/docker/daemon.json <<EOF
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
EOF
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# tee /etc/docker/daemon.json <<EOF
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
EOF
{
"exec-opts": ["native.cgroupdriver=systemd"],
"log-driver": "json-file",
"log-opts": {
"max-size": "100m"
},
"storage-driver": "overlay2"
}
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```


- Writes kernel parameters required by Kubernetes to a sysctl configuration file.

Parameters ensure network packet routing between pods, bridge network traffic pass-through, and IP forwarding.

```
tee /etc/sysctl.d/kubernetes.conf<<EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# tee /etc/sysctl.d/kubernetes.conf<<EOF
> net.bridge.bridge-nf-call-ip6tables = 1
> net.bridge.bridge-nf-call-iptables = 1
> net.ipv4.ip_forward = 1
> EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Reload systemd configuration and restart the containerd service.

```
systemctl daemon-reload
systemctl restart containerd
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# systemctl daemon-reload
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# systemctl restart containerd
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Initialize a Kubernetes master node with a specified pod network IP range.

```
kubeadm init --pod-network-cidr=10.244.0.0/16
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubeadm init --pod-network-cidr=10.244.0.0/16
I0908 09:16:21.124956   53797 version.go:255] remote version is much newer: v1.28.1; falling back to: stable-1.22
[init] Using Kubernetes version: v1.22.17
[preflight] Running pre-flight checks
        [WARNING SystemVerification]: this Docker version is not on the list of validated versions: 24.0.5. Latest validated version: 20.10
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
```

- Set up local Kubernetes admin config and remove master node restrictions for pod scheduling

---

Taints and Tolerations in Kubernetes help dictate where pods should or shouldn't be scheduled on nodes.

```
rm ~/.kube/config
mkdir -p $HOME/.kube
cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
kubectl taint nodes --all node-role.kubernetes.io/master-
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# mkdir -p $HOME/.kube
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl taint nodes --all node-role.kubernetes.io/master-
node/ip-10-0-0-214 untainted
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Apply the Flannel networking configuration from its official GitHub repository.

```
kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master
/Documentation/kube-flannel.yml
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Download and install Helm 3, add the Falco security chart repository, update the repo list, and deploy Falco with ebf configurations.

```
kubectl get pods
curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
helm repo add falcosecurity https://falcosecurity.github.io/charts
helm repo update
helm install falco --set driver.kind=ebpf --set tty=true falcosecurity/falco
--version 3.6.2
```

Latest version not working on ubuntu instance we are using in lap due to file not found issue.

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl get pods
No resources found in default namespace.
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11715  100 11715    0     0  95175      0 --:--:-- --:--:-- --:--:-- 96024
Downloading https://get.helm.sh/helm-v3.12.3-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# helm repo add falcosecurity https://falcosecurity.github.io/charts
"falcosecurity" has been added to your repositories
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "falcosecurity" chart repository
Update Complete. ⎈Happy Helming!⎈
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# helm install falco --set driver.kind=ebpf --set tty=true falcosecurity/falco
NAME: falco
LAST DEPLOYED: Fri Sep  8 09:20:37 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Falco agents are spinning up on each node in your cluster. After a few
seconds, they are going to start monitoring your containers looking for
security issues.


No further action should be required.


Tip:
You can easily forward Falco events to Slack, Kafka, AWS Lambda and more with falcosidekick.
Full list of outputs: https://github.com/falcosecurity/charts/tree/master/falcosidekick.
You can enable its deployment with `--set falcosidekick.enabled=true` or in your values.yaml.
See: https://github.com/falcosecurity/charts/blob/master/falcosidekick/values.yaml for configuration values.
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Wait for falco pod to run.

```
kubectl get pods && sleep 30 && kubectl get pods
sleep 70 && kubectl get pods
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl get pods && sleep 30 && kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
falco-z9h57   1/2     Running   0          67s
NAME          READY   STATUS    RESTARTS   AGE
falco-z9h57   2/2     Running   0          97s
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# sleep 70 && kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE                              sleep 70 && kubectl get pods
falco-z9h57   2/2     Running   0          5m18s
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

- Run nginx pod for testing the malicious activity.

```
kubectl run nginx-pod --image=nginx --restart=Never
kubectl get pods && sleep 20
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl run nginx-pod --image=nginx --restart=Never
pod/nginx-pod created
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl get pods && sleep 20
NAME          READY   STATUS              RESTARTS   AGE
falco-z9h57   2/2     Running             0          6m31s
nginx-pod     0/1     ContainerCreating   0          0s
```

- Check the pods status.

```
kubectl get pods
```



```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
falco-z9h57   2/2     Running   0          7m22s
nginx-pod     1/1     Running   0          51s
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

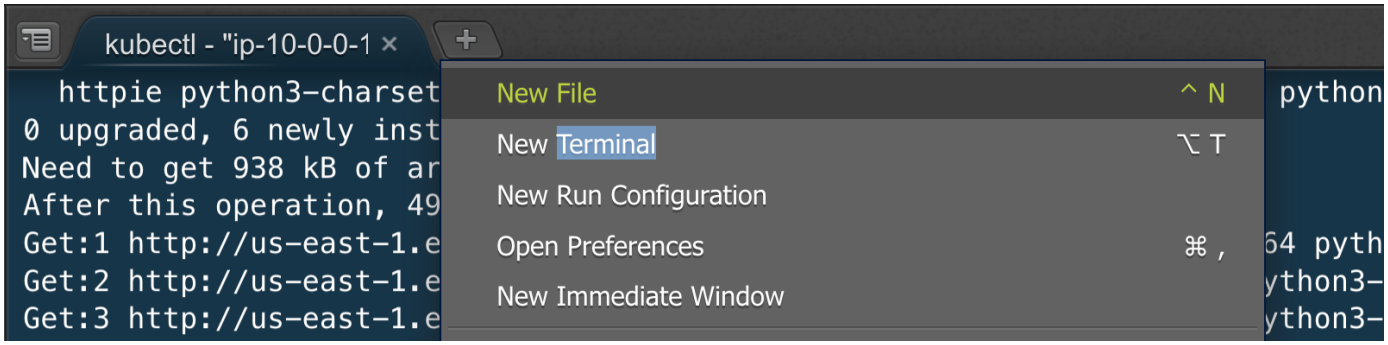
- Try accessing `/etc/passwd`, this is mimicking malicious activity.

```
kubectl exec -it nginx-pod -- cat /etc/shadow
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl exec -it nginx-pod -- cat /etc/shadow
root:*:19604:0:99999:7:::
daemon:*:19604:0:99999:7:::
bin:*:19604:0:99999:7:::
sys:*:19604:0:99999:7:::
sync:*:19604:0:99999:7:::
games:*:19604:0:99999:7:::
man:*:19604:0:99999:7:::
lp:*:19604:0:99999:7:::
mail:*:19604:0:99999:7:::
news:*:19604:0:99999:7:::
uucp:*:19604:0:99999:7:::
proxy:*:19604:0:99999:7:::
www-data:*:19604:0:99999:7:::
backup:*:19604:0:99999:7:::
list:*:19604:0:99999:7:::
irc:*:19604:0:99999:7:::
_apt:*:19604:0:99999:7:::
nobody:*:19604:0:99999:7:::
nginx:!:19607::::::
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

Open Another Terminal

- Click on `+` icon, then select `new terminal` to open new terminal.



- In another terminal, filter the logs to include only those lines that contain the word "shadow". This command line will continuously stream logs from the first "falco-" pod and filter for entries containing "shadow".

```
kubectl get pods | grep falco- | awk '{if ($3 == "Running") print $1}' | awk
'NR==1{print $1}' | xargs -I{} kubectl logs -f {} | grep shadow
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# kubectl get pods | grep falco- | awk '{if ($3 == "Running") print $1}' | awk 'N
R==1{print $1}' | xargs -I{} kubectl logs -f {} |grep shadow
Defaulted container "falco" out of: falco, falcoctl-artifact-follow, falco-driver-loader (init), falcoctl-artifact-install (init)
09:27:10.991705467: Error File below /etc opened for writing (user=root user_loginuid=-1 command=dockerd -H fd:// --containerd=/run/containerd/containerd.sock
pid=841 parent=systemd pcmdline=systemd --system --deserialize 39 file=/etc/gshadow program=dockerd gparent=<NA> ggparent=<NA> gggparent=<NA> container_id=host
 image=<NA>) k8s.ns=<NA> k8s.pod=<NA> container=host
09:27:11.026661798: Error File below /etc opened for writing (user=root user_loginuid=-1 command=dockerd -H fd:// --containerd=/run/containerd/containerd.sock
pid=841 parent=systemd pcmdline=systemd --system --deserialize 39 file=/etc/shadow program=dockerd gparent=<NA> ggparent=<NA> gggparent=<NA> container_id=host
image=<NA>) k8s.ns=<NA> k8s.pod=<NA> container=host
09:27:12.452156093: Error File below a known binary directory opened for writing (user=root user_loginuid=-1 command=dockerd -H fd:// --containerd=/run/contain
erd/containerd.sock pid=841 file=/usr/sbin/shadowconfig parent=systemd pcmdline=systemd --system --deserialize 39 gparent=<NA> container_id=host image=<NA>) k8
s.ns=<NA> k8s.pod=<NA> container=host
09:27:13.366804886: Error File below /etc opened for writing (user=root user_loginuid=-1 command=dockerd -H fd:// --containerd=/run/containerd/containerd.sock
pid=841 parent=systemd pcmdline=systemd --system --deserialize 39 file=/etc/gshadow program=dockerd gparent=<NA> ggparent=<NA> gggparent=<NA> container_id=host
 image=<NA>) k8s.ns=<NA> k8s.pod=<NA> container=host
09:27:13.367117336: Error File below /etc opened for writing (user=root user_loginuid=-1 command=dockerd -H fd:// --containerd=/run/containerd/containerd.sock
pid=841 parent=systemd pcmdline=systemd --system --deserialize 39 file=/etc/gshadow- program=dockerd gparent=<NA> ggparent=<NA> gggparent=<NA> container_id=hos
t image=<NA>) k8s.ns=<NA> k8s.pod=<NA> container=host
09:27:13.373550479: Error File below /etc opened for writing (user=root user_loginuid=-1 command=dockerd -H fd:// --containerd=/run/containerd/containerd.sock
pid=841 parent=systemd pcmdline=systemd --system --deserialize 39 file=/etc/shadow program=dockerd gparent=<NA> ggparent=<NA> gggparent=<NA> container_id=host
image=<NA>) k8s.ns=<NA> k8s.pod=<NA> container=host
09:27:13.373722101: Error File below /etc opened for writing (user=root user_loginuid=-1 command=dockerd -H fd:// --containerd=/run/containerd/containerd.sock
pid=841 parent=systemd pcmdline=systemd --system --deserialize 39 file=/etc/shadow- program=dockerd gparent=<NA> ggparent=<NA> gggparent=<NA> container_id=host
 image=<NA>) k8s.ns=<NA> k8s.pod=<NA> container=host
09:28:22.366951843: Warning Sensitive file opened for reading by non-trusted program (user=root user_loginuid=-1 program=cat command=cat /etc/shadow pid=62655
file=/etc/shadow parent=<NA> gparent=<NA> ggparent=<NA> gggparent=<NA> container_id=edfc19be920b image=nginx) k8s.ns=default k8s.pod=nginx-pod container=edfc19
be920b
```

Cleanup

- Uninstall Helm release for falco.

```
helm uninstall falco
```

```
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4# helm uninstall falco
release "falco" uninstalled
root@ip-10-0-0-214:/home/ubuntu/ workspace/course/8_detection/falco-workshop-4#
```

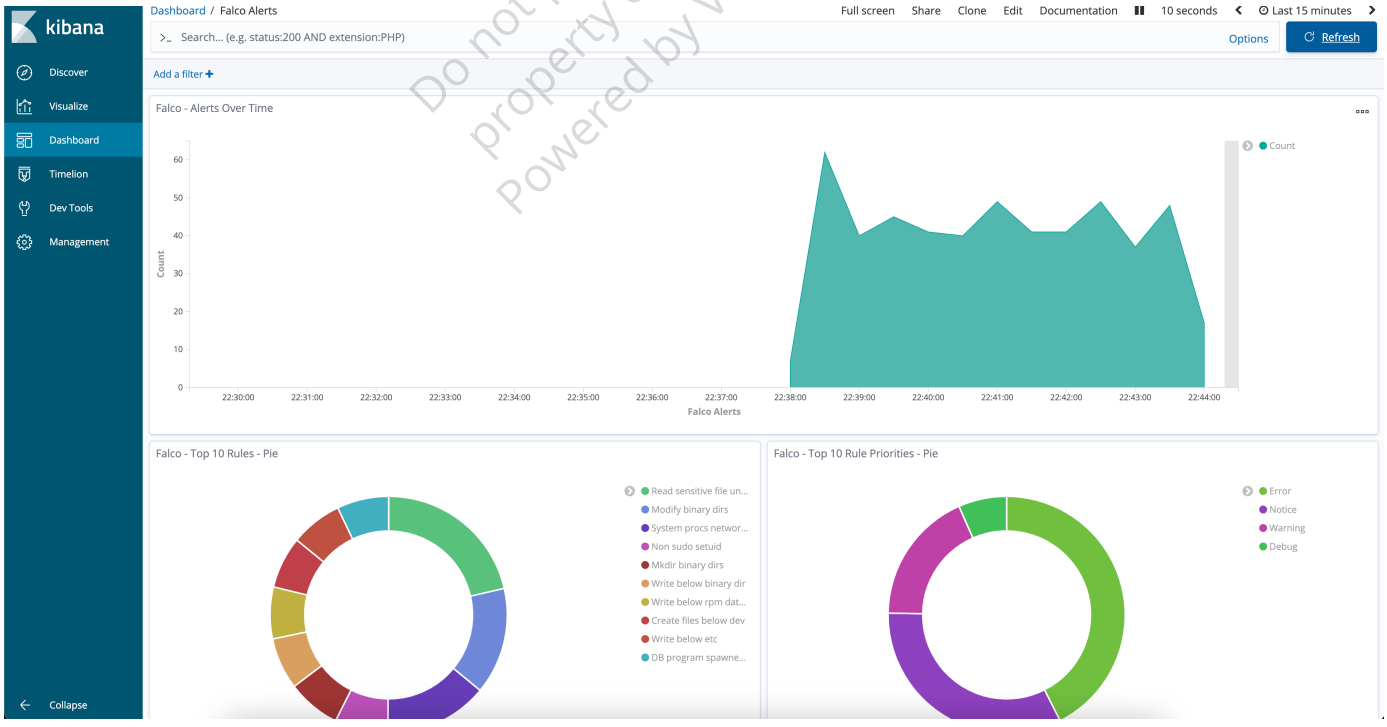
Demo: EFK Logging and Monitoring

- Kubernetes security logging primarily focuses on orchestrator events.
- The Kubernetes documentation provides a good starting point for auditing events of the Kubernetes API.
- Using Sysdig Falco and Fluentd can provide a more complete Kubernetes security logging solution, giving you the ability to see abnormal activity inside application and kube-system containers.

Using sysdig event generator to generate the traffic

- Access The EFK Dashboard

Note: This is generated during the training.



Live Dashboard

Reference:

- https://falco.org/docs/install-operate/third-party/learning/
- https://falco.org/docs/getting-started/falco-kubernetes-quickstart/
- https://falco.org/docs/install-operate/installation/