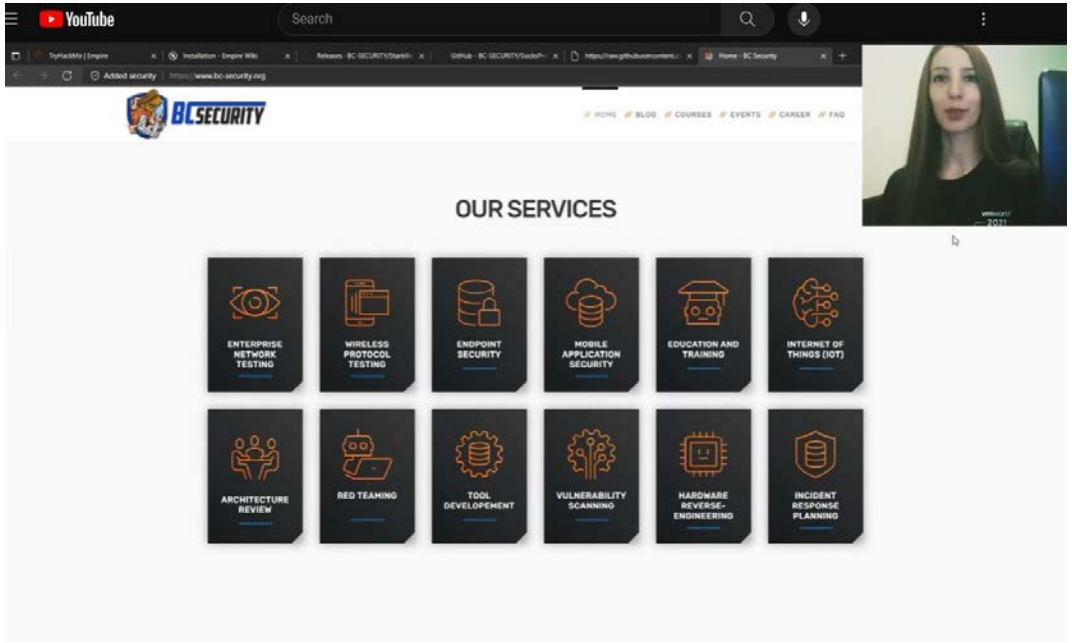
sysdig

Runtime Kubernetes Security: Hands-On Threat Detection with Falcosidekick

ALEKSANDRA DROBNJAK
CUSTOMER SOLUTIONS ENGINEER
SYSDIG

whoami



AWS community builders

Cisco Insider Champion 2024

Cloud Security Engineer · Google Cloud Certified · Professional

CERTIFIED kubernetes SECURITY SPECIALIST

https://www.linkedin.com/in/aleksandra-drobnjak

Cloud and Kubernetes Jobs

amazonjobs | kubernetes

- Systems, Quality, and Security Engineering (7)
- Business and Merchant Development (6)
- Sales, Advertising, and Account Management (4)
- Marketing and PR (3)
- Machine Learning Science (1)
- Project/Programme/Product Management--

Product Marketing Manager - Serverless and Kubernetes, Serverless and Kubernetes PMM

Posted August 12, 2024
(Updated 25 days ago)

Santa Clara, CA, USA +1 other locations | Job ID: 2730136

Basic qualifications:

- 4+ years of professional non-internship marketing experience
- Experience in partnership and product marketing
- Experience using data and metrics to measure impact and determine improvements
- Experience or background in software, SaaS or DevOps or 2+ years of post MBA

...Read more

Cisco Careers

Search jobs | Careers home | Emerging Talent | Events | Stay In Touch

< Back to search results

Kubernetes Security Engineer

sysdig | Products | Solutions | Open Source | Why Sysdig | Resources | Log In | GET DEMO →

ZARAGOZA
Join our Talent Community! MeetUp Event, October 17 at Sysdig Zaragoza office
ENGINEERING →

FLEXIBLE - ITALY
Sr. Software Engineer
ENGINEERING →

FLEXIBLE - CA
Sr. Security Engineer
ENGINEERING →

Google Careers | Teams | Locations | Benefits | Jobs | Students

76 jobs matched | Clear filters

What do you want to do?
kubernetes

- Locations
- Experience
- Skills & qualifications
- Degree

Software Engineer, Google Kubernetes Engine and Networking

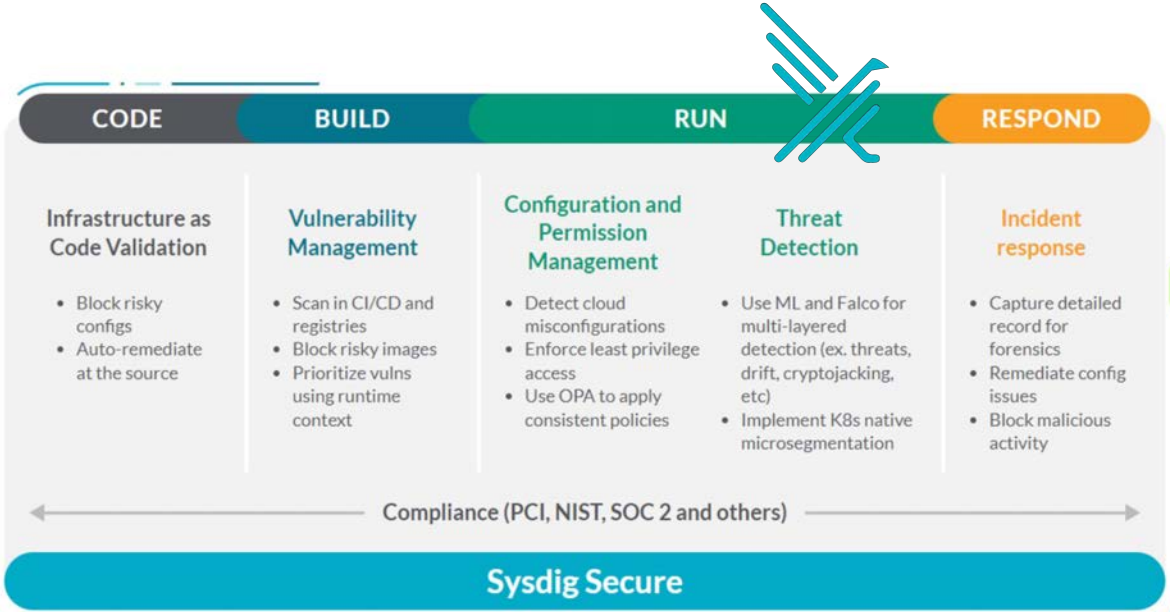
Google | Warsaw, Poland | Mid

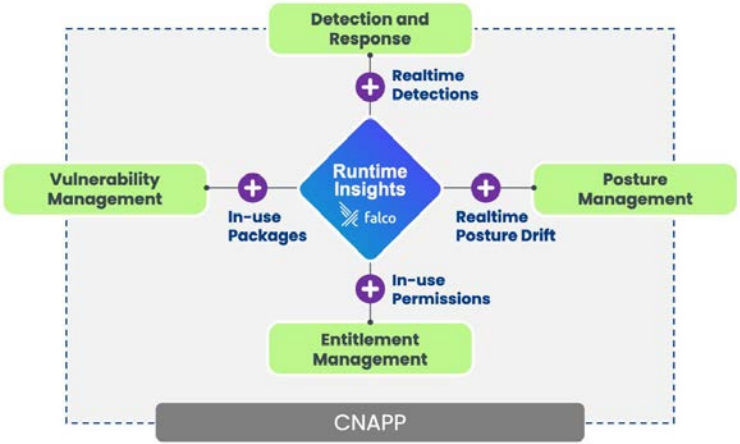
Minimum qualifications

- Bachelor's degree or equivalent practical experience.
- 2 years of experience with software development in one or more programming languages, or 1 year of experience with an advanced degree.
- 2 years of experience with data structures or algorithms.
- 2 years of experience with developing large-scale infrastructure, distributed systems or networks, or experience with compute technologies, storage or hardware architecture.

Learn more

Cloud-Native Application Protection Platform

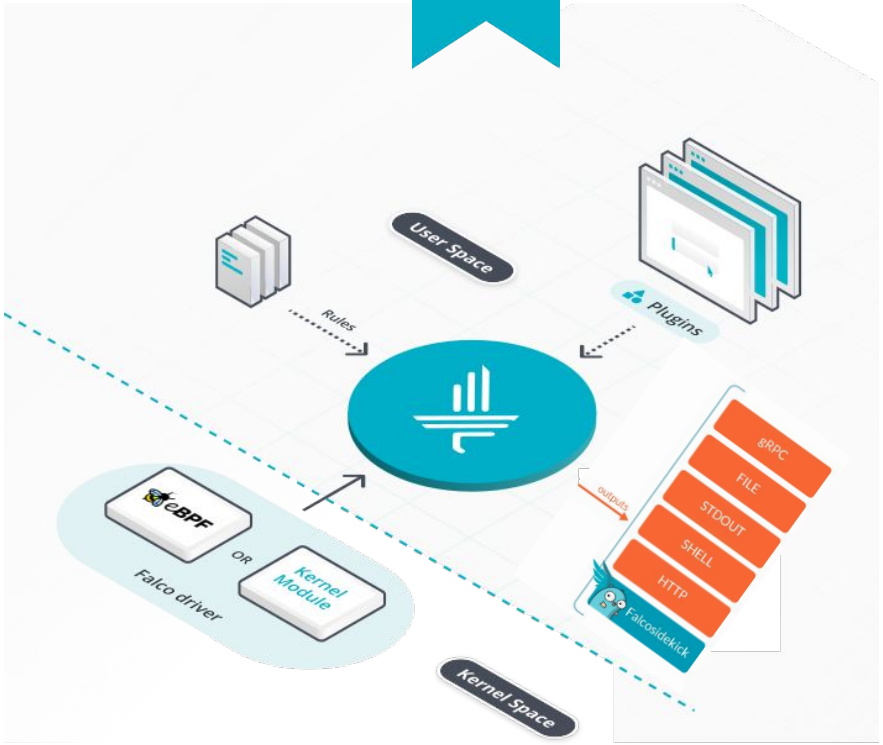


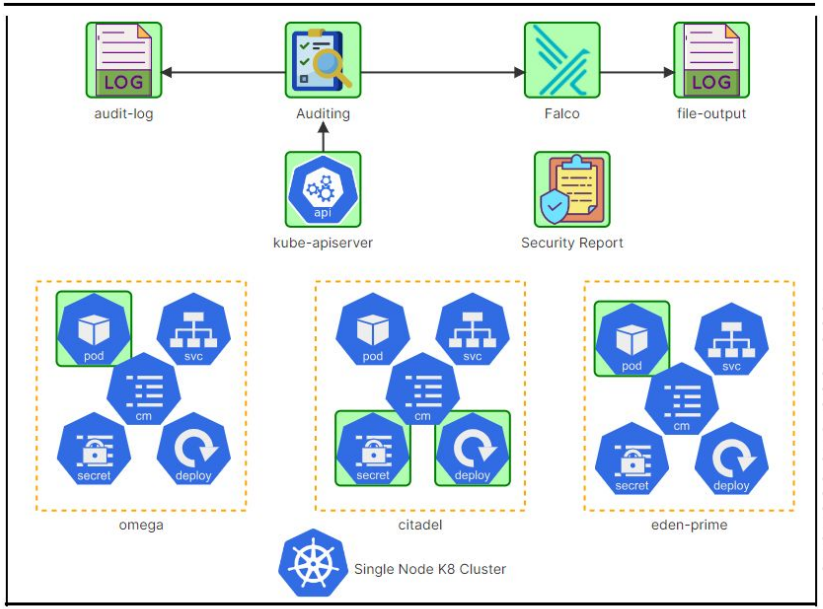



```
controlplane $ kubectl logs -f --tail=0 -n falco -c falco -l app.kubernetes.io/name=falco | grep 'Potentially malicious Python script'
18:51:00.000847841: Warning Potentially malicious Python script encoded on command line (proc.cmdline=bash -c filename='/etc/rc.common';if [ ! -f $filename ];then sudo touch $filename;else sudo cp $filename /etc/rc.common.o
riginal;fi; printf '%s\\n' '#!/bin/bash' | sudo tee /etc/rc.common; echo "python3 -c \"import os, base64;exec(base64.b64decode('aW1wb3J0IG9zCm9zLnBvcGVuKCdlY2hvIGF0b21pYyB0ZXN0IGZvciBtb2RpZnlpbmcgcmMuY29tbW9uID4gL3RtcC9UMTA
zNy4wMDQucmMuY29tbW9uJykK'))\"" | sudo tee -a /etc/rc.common; printf '%s\\n' 'exit 0' | sudo tee -a /etc/rc.common; sudo chmod +x /etc/rc.common user.name=root proc.name=bash proc.pname=Pipeline evt.type=execve gparent=bash
 ggparent=containerd-shim gggparent=systemd evt.res=SUCCESS container.id=125a9adb5cfd container.name=atomicred file=<NA>) k8s.ns=atomic-red k8s.pod=atomicred-6f75785845-7bzxv container=125a9adb5cfd
^C
controlplane $ kubectl logs -f --tail=0 -n falco -c falco -l app.kubernetes.io/name=falco | grep 'Log files were tampered'
18:56:39.923179011: Warning Log files were tampered (user=root user_loginuid=-1 command=sh -c cat /dev/null > /var/log/messages #truncating the file to zero bytes; cat /dev/zero > /var/lol/messages #log file filled with nul
l bytes(zeros) pid=53507 file=/var/log/messages container_id=125a9adb5cfd image=docker.io/issif/atomic-red) k8s.ns=atomic-red k8s.pod=atomicred-6f75785845-7bzxv container=125a9adb5cfd
^C
controlplane $ kubectl logs -f --tail=0 -n falco -c falco -l app.kubernetes.io/name=falco | grep 'Linux Kernel Module injection from container detected'
18:57:46.301361064: Warning Linux Kernel Module injection from container detected (user=root uid=0 user_loginuid=-1 process_name=insmod parent_process_name=sudo parent_exepath=/usr/bin/sudo sh /usr/bin/sh module=/root/Atomi
cRedTeam/atomics/T1014/bin/T1014.ko k8s.ns=atomic-red k8s.pod=atomicred-6f75785845-7bzxv container=125a9adb5cfd image=docker.io/issif/atomic-red:latest res=<NA> syscall=finit_module)
```

Falco

Falco
CNCF 2024
GRADUATED





KodeKloud CKS Scenario 4

Task

There are a number of Kubernetes objects created inside the `omega`, `citadel` and `eden-prime` namespaces. However, several suspicious/abnormal operations have been observed in these namespaces!.

For example, in the `citadel` namespace, the application called `webapp-color` is constantly changing! You can see this for yourself by clicking on the `citadel-webapp` link and refreshing the page every 30 seconds. Similarly there are other issues with several other objects in other namespaces.

To understand what's causing these anomalies, you would be required to configure `auditing` in Kubernetes and make use of the `Falco` tool.

Inspect the issues in detail by clicking on the icons of the interactive architecture diagram on the right and complete the tasks to secure the cluster. Once done click on the `Check` button to validate your work.

Check

Falco Rules

Falco

Type: all | list | macro | rule

Priority: all | DEBUG | INFO | NOTICE | WARNING | ERROR | CRITICAL | ALERT

Source: all | okta | aws_cloudtrail | github | gcp_auditlog | syscalls | k8s_audit

Maturity: all | stable | incubating | sandbox | deprecated

Status: all | disabled | enabled

Show 100 entries

Type	Priority	Source	Name	File
rule	WARNING	syscalls	Clear Log Activities	falco_rules.yaml
rule	WARNING	syscalls	Create Hardlink Over Sensitive Files	falco_rules.yaml
rule	WARNING	syscalls	Create Symlink Over Sensitive Files	falco_rules.yaml
rule	WARNING	syscalls	Debugfs Launched in Privileged Container	falco_rules.yaml
rule	WARNING	syscalls	Directory traversal monitored file read	falco_rules.yaml
rule	WARNING	syscalls	Execution from /dev/shm	falco_rules.yaml
rule	WARNING	syscalls	Find AWS Credentials	falco_rules.yaml
rule	WARNING	syscalls	Linux Kernel Module Injection Detected	falco_rules.yaml
rule	WARNING	syscalls	Netcat Remote Code Execution in Container	falco_rules.yaml
rule	WARNING	syscalls	PTRACE attached to process	falco_rules.yaml
rule	WARNING	syscalls	Read sensitive file trusted after startup	falco_rules.yaml
rule	WARNING	syscalls	Read sensitive file untrusted	falco_rules.yaml
rule	WARNING	syscalls	Remove Bulk Data from Disk	falco_rules.yaml
rule	WARNING	syscalls	Search Private Keys or Passwords	falco_rules.yaml

Type: rule

Priority: WARNING

Name: 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 https://github.com/swisskyrepo/PayloadsAllTheThings/. These programs are of higher relevance as they are commonly installed on UNIX-like operating systems. Can fire in combination with the "Redirect STDOUT/STDIN to Network Connection in Container" rule as it utilizes a different evt.type.

Source: syscalls

Condition:

```
spawned_process and container and ((proc.name = "nc" and (proc.cmdline contains " -
e" or
                                proc.cmdline contains " -e")) 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)
```

Status: stable

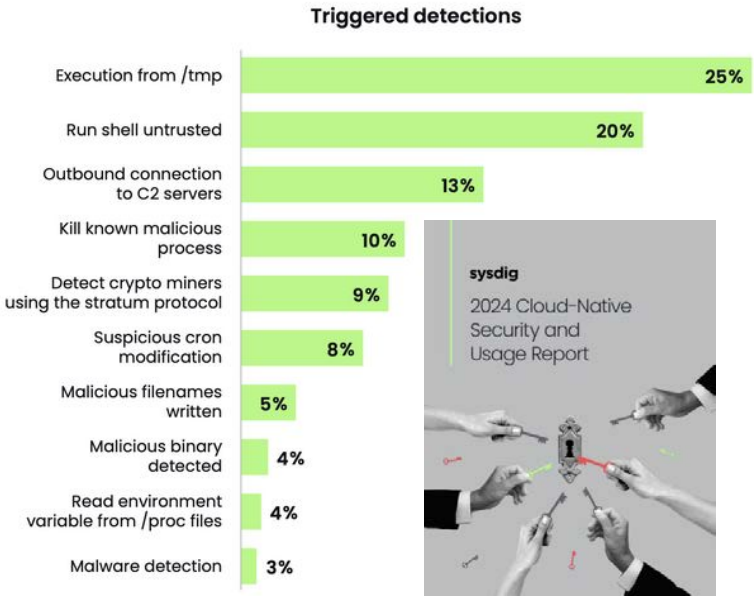
Status: enabled

Required engine version: 0.31.0

Tags: maturity_stable | container | network | process | mitre_execution | T1059

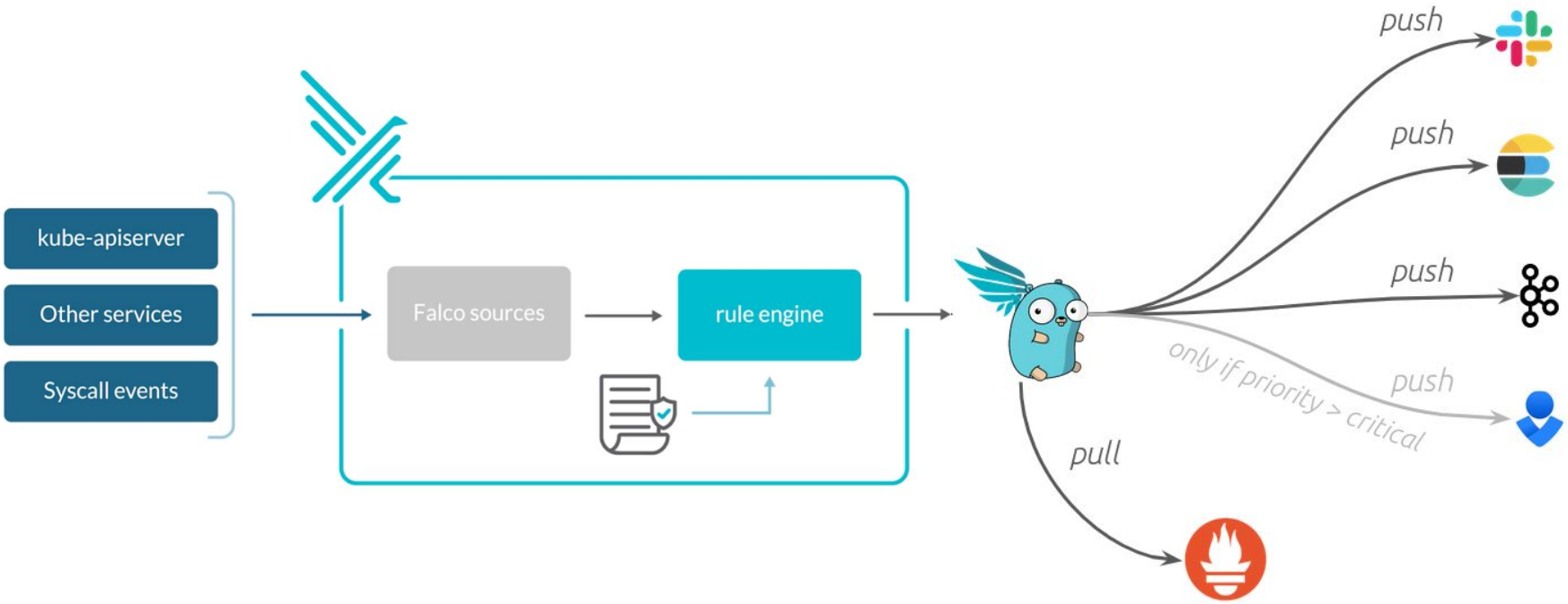
Depends on:

- macro spawned_process
- macro container
- macro spawned_process
- macro container



https://thomas.labarussias.fr/falco-rules-explorer

Falcosidekick

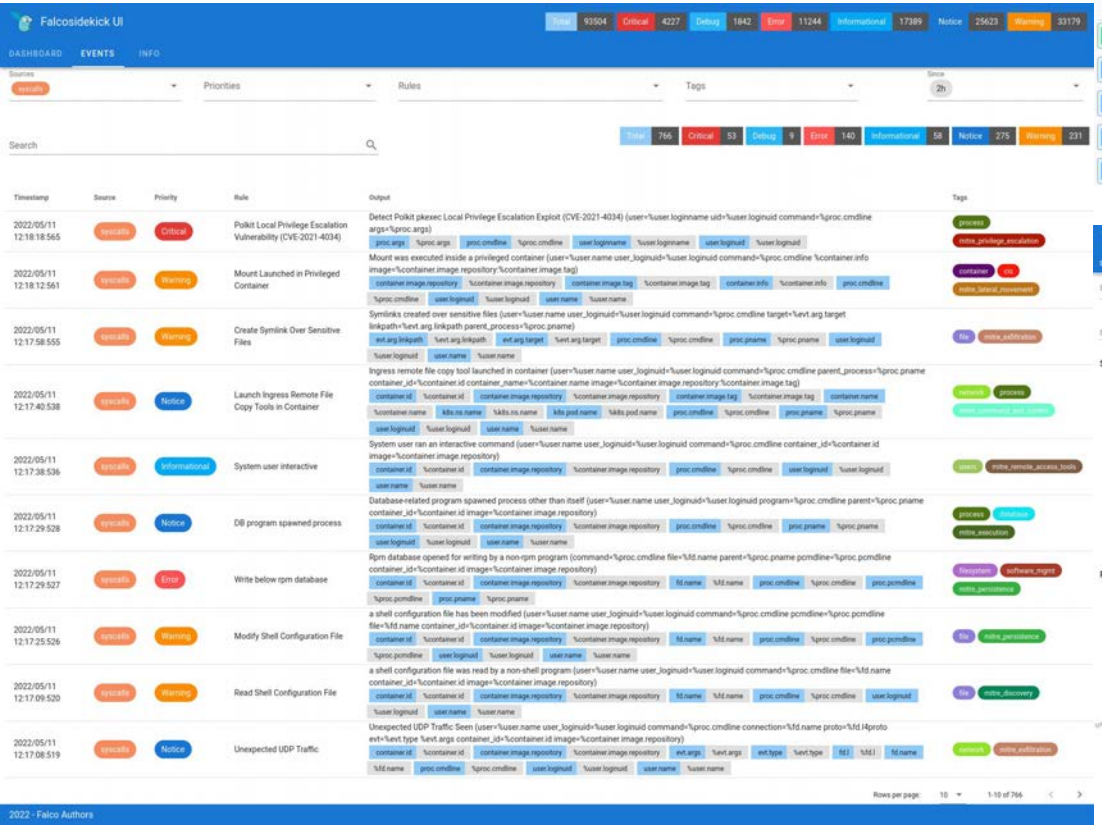


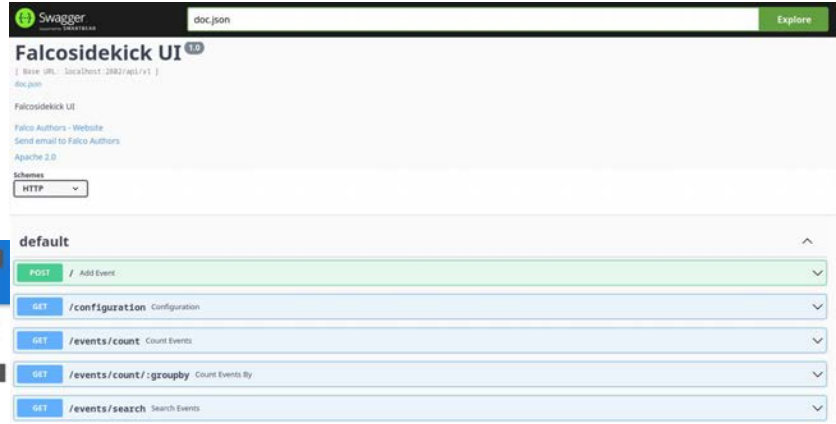
Events

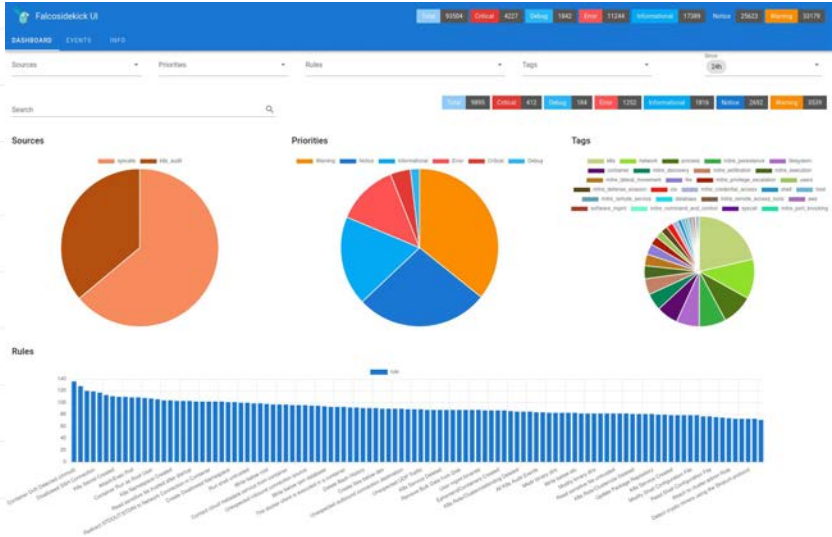
Rules

Alerts

Falcosidekick-ui







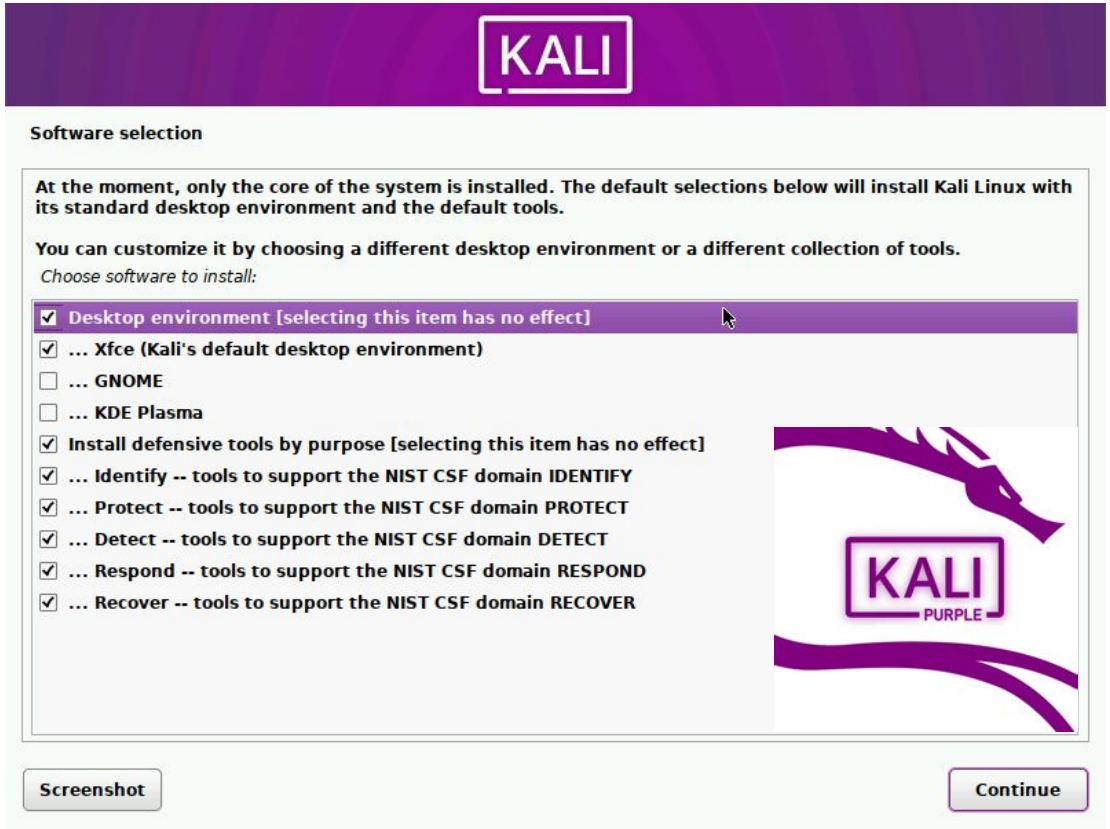
Lab Environment

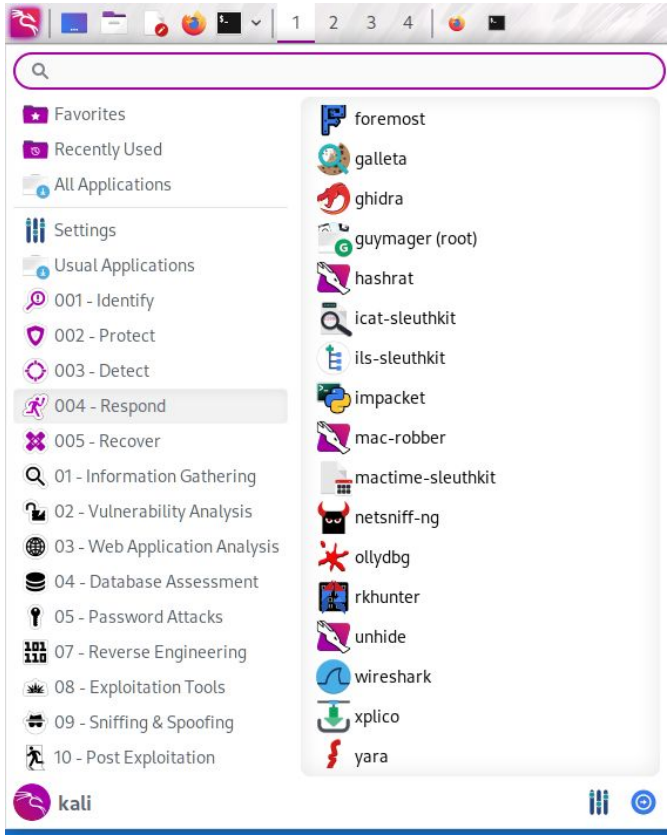
VM Download

https://drive.google.com/drive/folders/1FthEiIja5XKshE2EdfEsF7aTU_vz8QdE

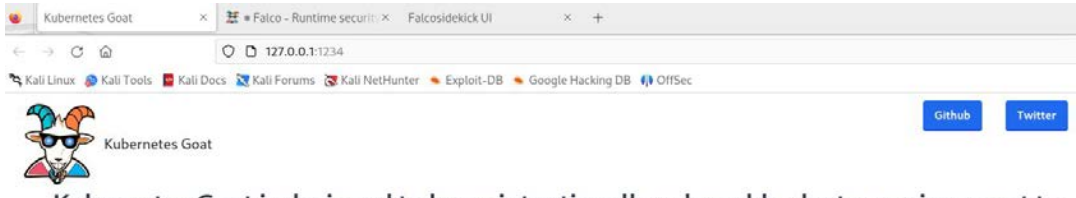
Kali Purple

Username: sidekick
Password: sidekick



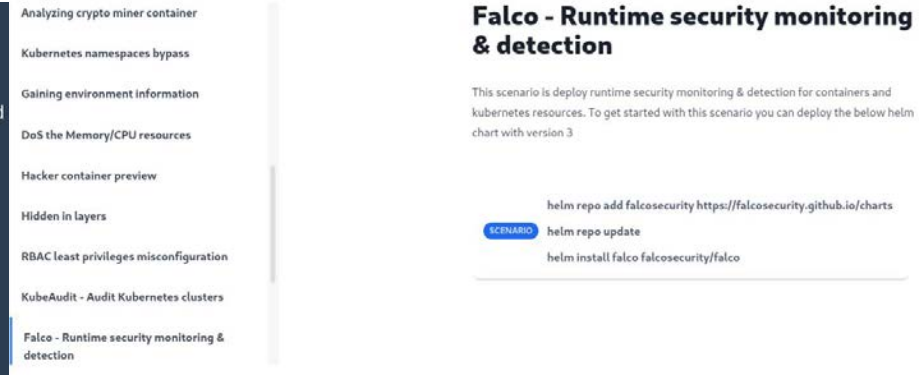


Kubernetes Goat




```
$ kubectl get pods

NAME                                          READY   STATUS
batch-check-job-brc55                         0/1     Completed
build-code-deployment-7cbd74ccdf-j5nz2        1/1     Running
health-check-deployment-7cd4d9ccd5-bwrkd      1/1     Running
hidden-in-layers-qk7sj                        1/1     Running
internal-proxy-deployment-6b897b7658-mcw2v    2/2     Running
kubernetes-goat-home-deployment-6676b97f8f-vv7gc  1/1  Running
metadata-db-b686dcff9-m9grt                   1/1     Running
poor-registry-deployment-7d66bf854f-kth5b     1/1     Running
system-monitor-deployment-669cd6459c-l8dcp    1/1     Running
```



Kubenomicon

Microsoft Tactics

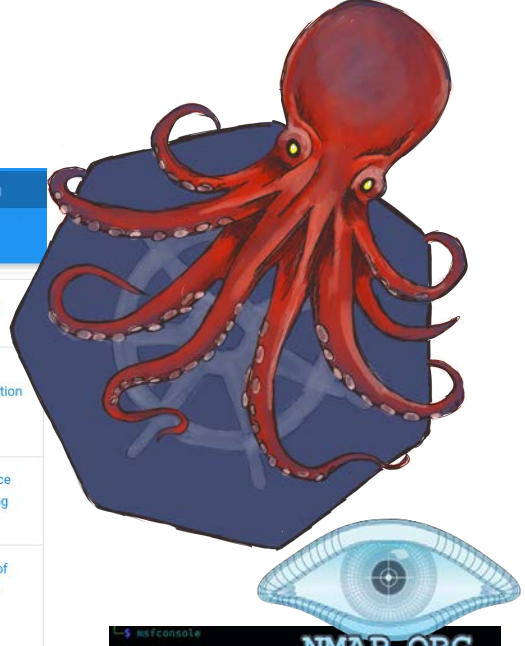
Search

Tactics Mitigations About

Tactics

- Initial Access
- Execution
- Persistence
- Privilege Escalation
- Defense Evasion
- Credential Access
- Discovery
- Lateral Movement
- Collection
- Impact

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Impact
Using cloud credentials	Exec into container	Backdoor container	Privileged container	Clear container logs	List K8S secrets	Access Kubernetes API server	Access cloud resources	Images from a private registry	Data destruction
Compromised image In registry	bash/cmd inside container	Writable hostPath mount	Cluster-admin binding	Delete K8S events	Mount service principal	Access Kubelet API	Container service account	Collecting data from pod	Resource hijacking
Kubeconfig file	New container	Kubernetes CronJob	hostPath mount	Pod / container name similarity	Container service account	Network mapping	Cluster internal networking		Denial of service
Application vulnerability	Application exploit (RCE)	Malicious admission controller	Access cloud resources	Connect from proxy server	Application credentials in configuration files	Exposed sensitive interfaces	Application credentials in configuration files		
Exposed sensitive interfaces	SSH server running inside container	Container service account			Access managed identity credentials	Instance Metadata API	Writable hostPath mount		
	Sidecar injection	Static pods			Malicious admission controller		CoreDNS poisoning		
							ARP poisoning and IP spoofing		



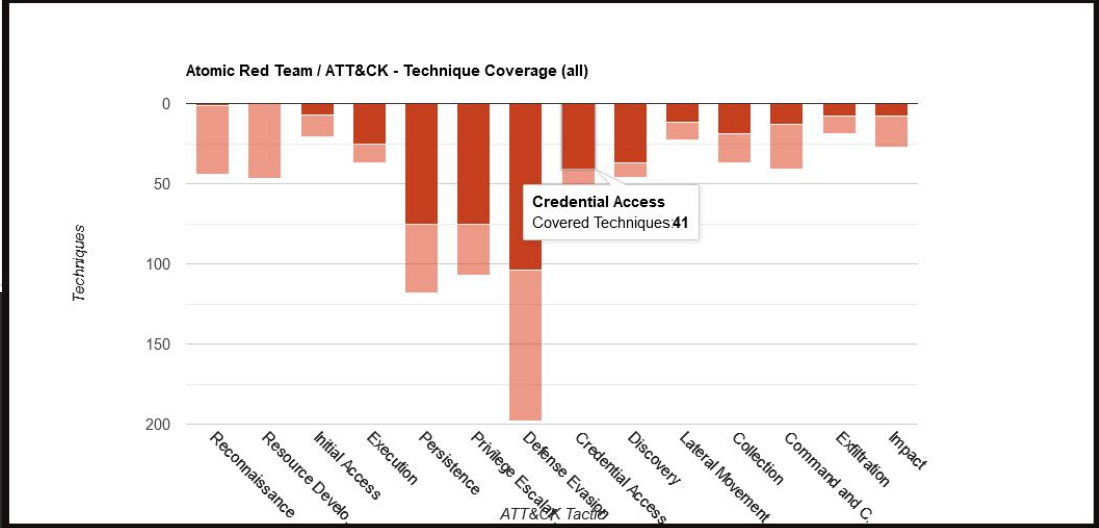
Atomic Red Team

All Platforms

- All Platforms
- macOS
- Windows
- Linux
- Google Workspace
- Azure Ad
- Office 365
- IaaS
- SaaS
- Containers

Atomic Red Team has tests for 204 of the 414 MITRE ATT&CK® Techniques for all of the platforms! (32%)
The community has created 1151 Atomic Tests for all of the platforms.

All Platforms




```
PS /root> Invoke-AtomicTest T1037.004 -ShowDetails
PathToAtomicsFolder = /root/AtomicRedTeam/atomics

[********BEGIN TEST********]
Technique: Boot or Logon Initialization Scripts: Rc.common T1037.004
Atomic Test Name: rc.common
Atomic Test Number: 2
Atomic Test GUID: c33f3d80-5f04-419b-a13a-854d1cbdbf3a
Description: Modify rc.common

Attack Commands:
Executor: bash
ElevationRequired: True
Command:
filename='/etc/rc.common';if [ ! -f $filename ];then sudo touch $filename;else sudo cp $filename /etc/rc.common.original;fi
printf '%s\n' '#!/bin/bash' | sudo tee /etc/rc.common
echo "python3 -c \"import os, base64;exec(base64.b64decode('aW1wb3J0IG9zCm9zLnBvcGVuKCdlY2hvIGF0b21pYyB0ZXN0IGZvciBtb2RpZnlpbmcgcmMuY29tbW9uID4gL3RtcC9UMTAzNy4wMDQucmMuY29tbW9uJykK'))\"" | sudo tee -a /etc/rc.common
printf '%s\n' 'exit 0' | sudo tee -a /etc/rc.common
sudo chmod +x /etc/rc.common

Cleanup Commands:
Command:
origfilename='/etc/rc.common.original';if [ ! -f $origfilename ];then sudo rm /etc/rc.common;else sudo cp $origfilename /etc/rc.common && sudo rm $origfilename;fi
[!!!!!!!!END TEST!!!!!!!]
```

TACTIC

- Collection
- Command And Control
- Credential Access
- Defense Evasion
- Discovery
- Execution
- Exfiltration
- Impact
- Initial Access
- Lateral Movement
- Persistence
- Privilege Escalation
- Reconnaissance

EXECUTORS

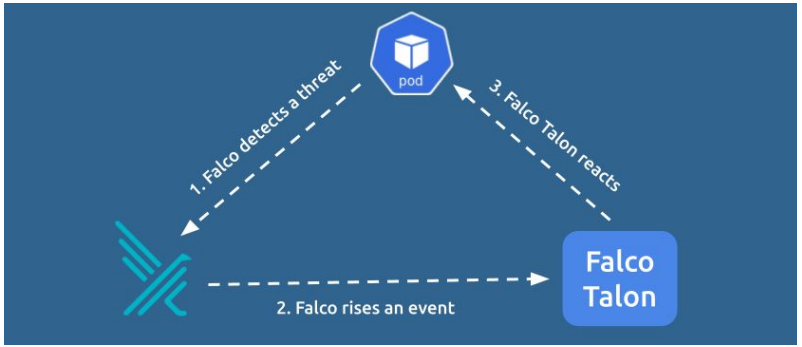
- bash
- command_prompt
- manual
- powershell
- sh

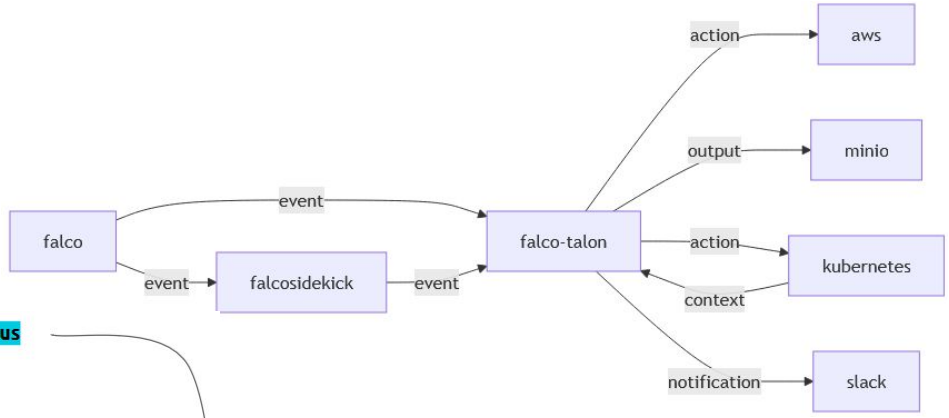
SUPPORTED PLATFORM

- azure-ad
- containers
- google-workspace
- iaas:aws
- iaas:azure
- iaas:gcp
- linux
- macos
- office-365
- windows

Lab Time

Next Steps: Falco Talon






```
- action: Disable outbound connections
  actionner: kubernetes:networkpolicy
  parameters:
    allow:
      - "192.168.1.0/24"
      - "172.17.0.0/16"
      - "10.0.0.0/32"

- action: Terminate Pod
  actionner: kubernetes:terminate

- rule: Suspicious outbound connection
  match:
    rules:
      - Unexpected outbound connection destination
  actions:
    - action: Disable outbound connections
    - action: Terminate Pod
      parameters:
        grace_period_seconds: 1
```

```
- action: Labelize Pod as Suspicious
  actionner: kubernetes:labelize
  parameters:
    labels:
      suspicious: true

- rule: Terminal shell in container
  match:
    rules:
      - Terminal shell in container
    output_fields:
      - k8s.ns.name!=kube-system, k8s.ns.name!=falco
  actions:
    - action: Labelize Pod as Suspicious
```

Stay in touch!

Falco

Join the Falco community

Join our Slack

Find us on GitHub

4.6K+

September 17, 2024

Rule Changes

- Updated Indicators of Compromise (IoC) rulesets with new findings.
- Added the following rules:
  - `Reverse Shell Detected`
  - `DNS Lookup for Reconnaissance Service Detected`
  - `DNS Lookup for Dynamic DNS Domain Detected`
  - `DNS Tunneling Activity Detected`
  - `Junk Data Padding Detected`

Courtesy of sysdig

5 hr 41 min

Falco 101

All you need to learn to get started with Falco

1 hr 27 min

Falco Plugins

Extending Falco to secure your cloud services

2 hr 00 min

Detecting a Cryptomining Malware attack with Falco and Prometheus

Learn how to detect Cryptominers with Falco and Prometheus

https://falco.org/training/

sysdig

SECURE
EVERY
SECOND