Falco

KubeCon | CloudNativeCon

North America 2024

Peak Innovation and Cloud Tweaks: Falco's Ongoing Runtime Security Development

Carlos Panato, Chainguard
Jason Dellaluce, Sysdig
Leonardo Grasso, Sysdig
Luca Guerra, Sysdig
Melissa Kilby, Apple

Thursday November 14, 2024 3:25pm - 4:00pm MST (Hyatt Regency | Level 2 | Salt Lake Ballroom C)

What's Falco?

The Security Camera for Your Cloud

- ➔ Cloud Native Runtime Security
- ➔ Detect Security Threats
- ➔ Deliver Real-Time Alerts

```
2022-04-07T12:51:08: Notice A shell was spawned in a container with an attached terminal (user=root
user_loginuid=-1 elastic_borg (id=a10bd3b1b2a8) shell=bash parent=<NA> cmdline=bash terminal=34816
container_id=a10bd3b1b2a8 image=ubuntu)
2022-04-07T12:51:41: Warning Netcat runs inside container that allows remote code execution
(user=root user_loginuid=-1 command=nc -e container_id=a10bd3b1b2a8 container_name=elastic_borg
image=ubuntu:latest)
```

Falco

A Flourishing Community and an Open Governance

Our Principles

- ➔ Open
- ➔ Respectful
- ➔ Diverse
- ➔ Transparent
- ➔ Vibrant

“The open nature of The Falco Project, its first principle, will never be a matter of change.” - Falco’s governance

Falco

How Falco Works

Monitoring, Evaluating, Alerting, Securing!

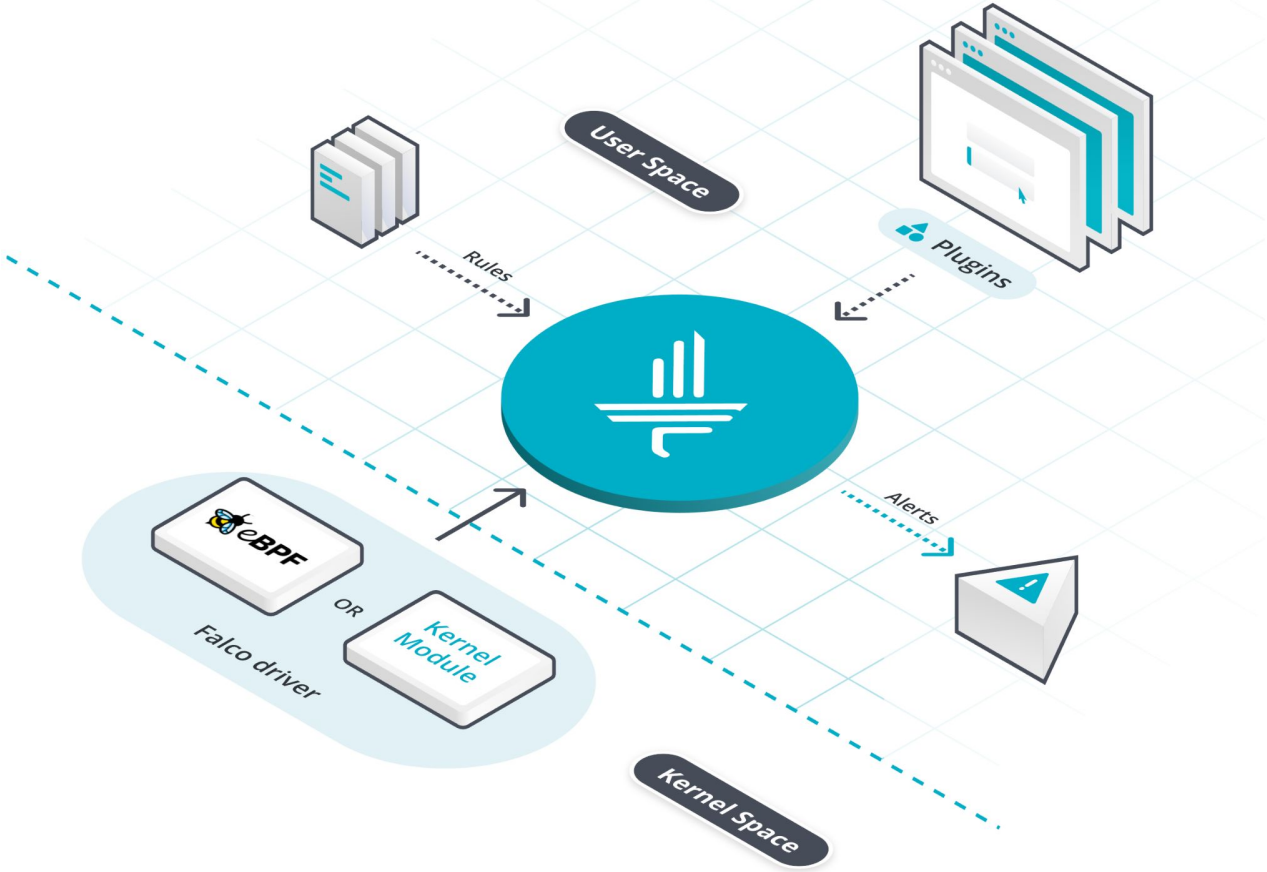
Kernel/Cloud Events Monitoring

👇

Rule Engine Evaluation

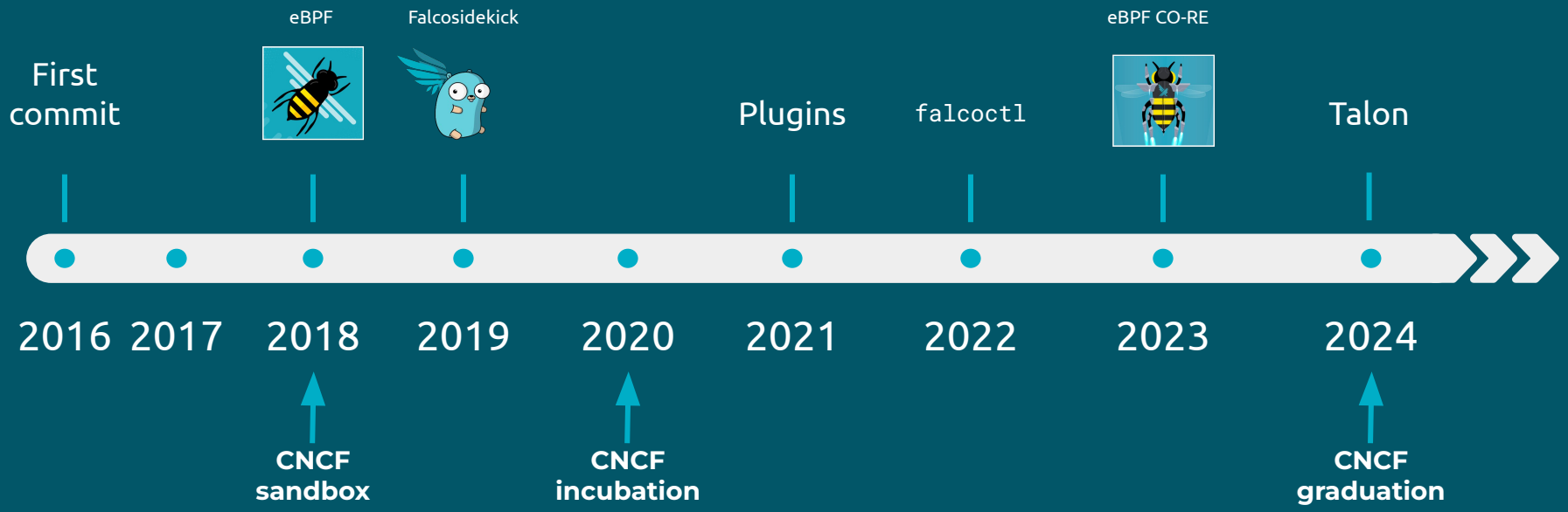
👇

Real-Time Alerts



The Dawn of Falco

From Inception to Cloud Native Excellence



Falco

What's Going On Right Now

Current Developments in Falco

- ➔ Enhanced Rule Handling
- ➔ Language Extensions and Operators
- ➔ Powerful New Plugins and SDKs
- ➔ Advanced Metrics
- ➔ Falco Talon

Falco

Falco

What’s New With Falco?

Falco Improvements

Detection Improvements: New Operators

Rule Selection and Output Customization

Driver Selection

Improved Detections - Transform operators

```
- rule: Search Private Keys or Passwords
  desc: [...]
  condition: >
    spawned_process and [...]
        (proc.name = "find" and (proc.args contains "id_rsa") [...] )
    [...]
```

```
proc.exepath = /bin/find or
proc.exepath = /usr/bin/find or
proc.exepath = /usr/local/bin/find
...
```

```
basename(proc.exepath) = find
```

Falco

Transform Operators

https://falco.org/docs/rules/conditions/#transform-operators

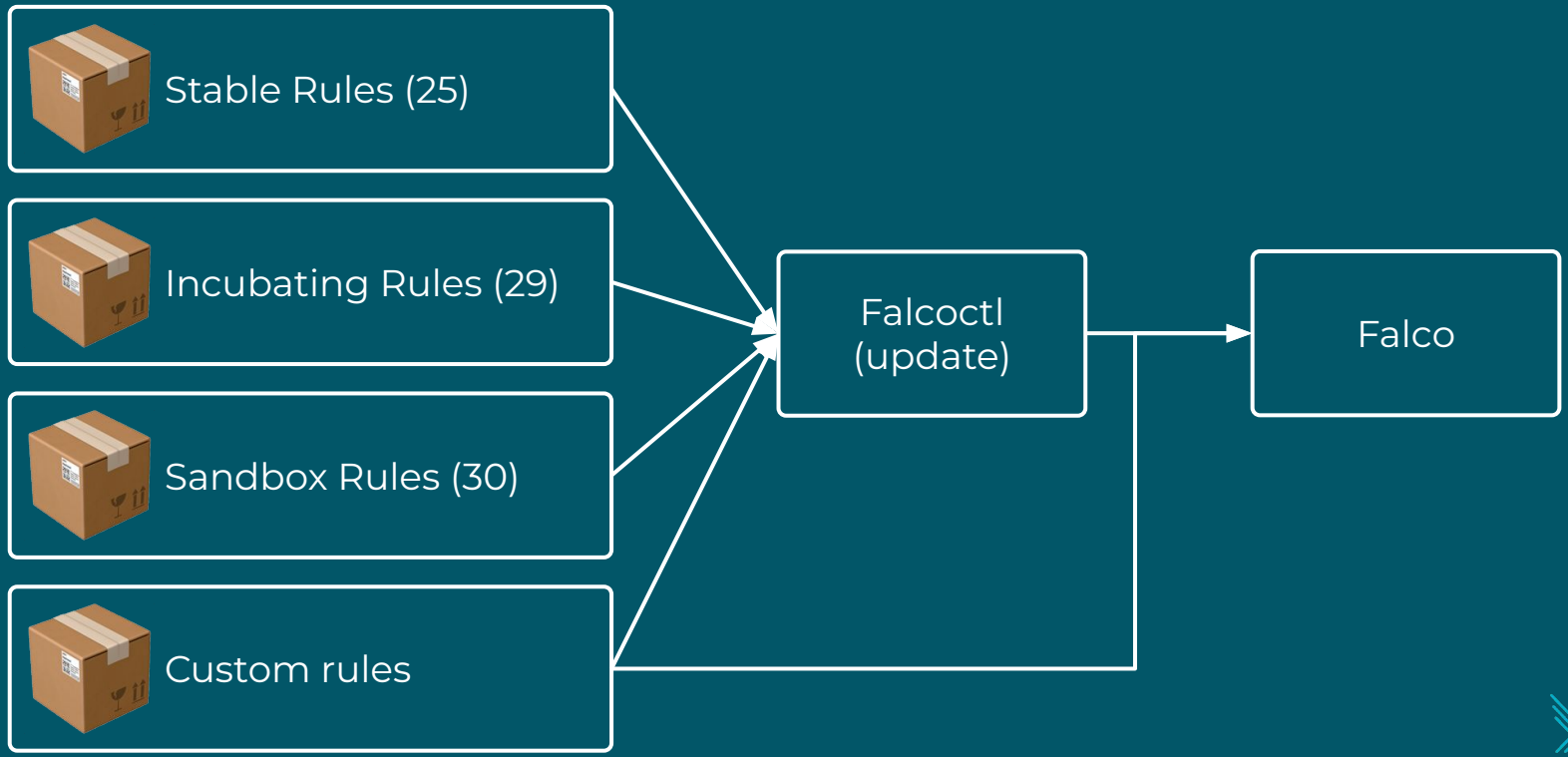
- **toupper()**
- **tolower()**
- **base64()**
- **basename()**
- **val()**

```
- rule: Process deleting its own executable
  desc: [...]
  condition: >
    evt.type = unlink and proc.exepath = val(fs.path.name)
```

Falco

Rule Selection and Output Customization

How do I select which rules to run? How do I customize my output?



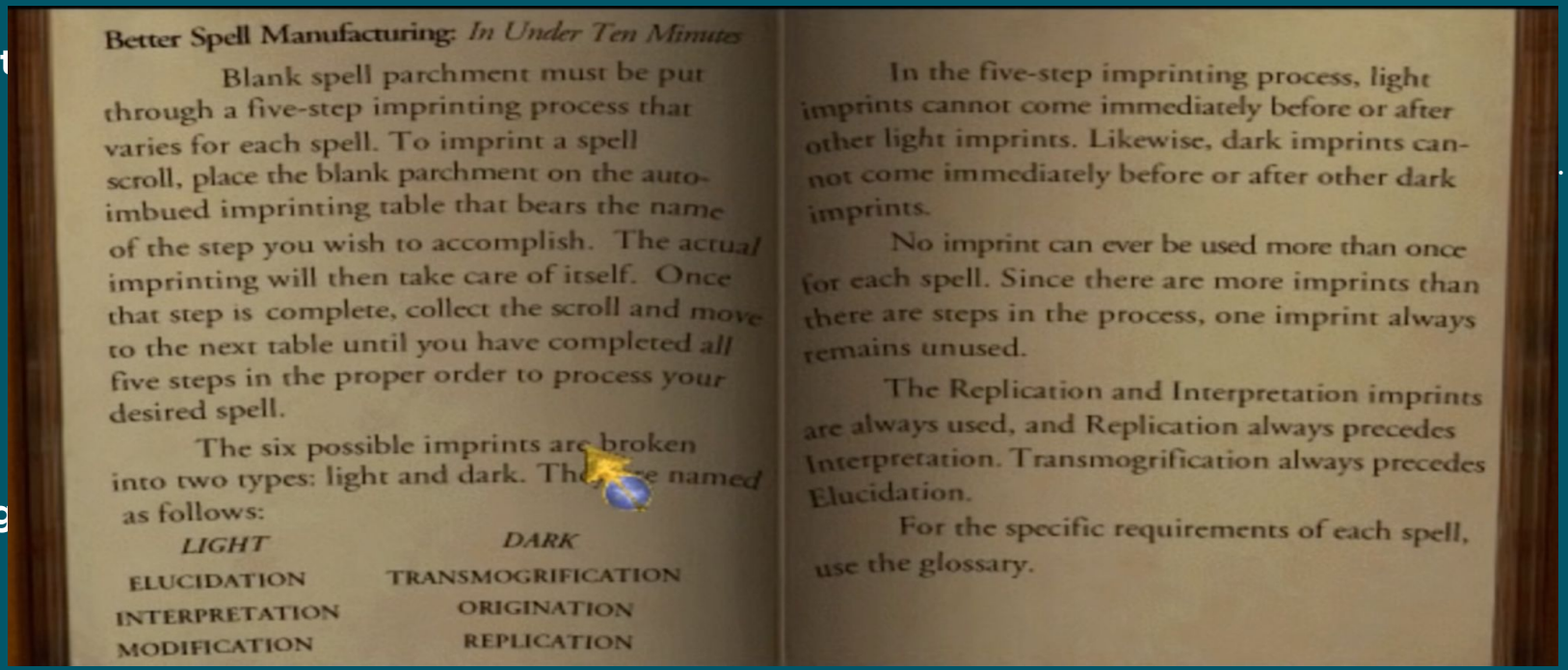
Falco

Rule Selection and Output Customization

Use t

-
-
-

Long



Better Spell Manufacturing: *In Under Ten Minutes*

Blank spell parchment must be put through a five-step imprinting process that varies for each spell. To imprint a spell scroll, place the blank parchment on the auto-imbued imprinting table that bears the name of the step you wish to accomplish. The actual imprinting will then take care of itself. Once that step is complete, collect the scroll and move to the next table until you have completed *all* five steps in the proper order to process your desired spell.

The six possible imprints are broken into two types: light and dark. The[illegible]e named as follows:

LIGHT	*DARK*
ELUCIDATION	TRANSMOGRIFICATION
INTERPRETATION	ORIGINATION
MODIFICATION	REPLICATION

In the five-step imprinting process, light imprints cannot come immediately before or after other light imprints. Likewise, dark imprints cannot come immediately before or after other dark imprints.

No imprint can ever be used more than once for each spell. Since there are more imprints than there are steps in the process, one imprint always remains unused.

The Replication and Interpretation imprints are always used, and Replication always precedes Interpretation. Transmogrification always precedes Elucidation.

For the specific requirements of each spell, use the glossary.

Falco

Rule Selection

Since Falco 0.38.0, in falco.yaml

```
rules:
    - disable:
        rule: “*”
    - enable:
        tag: persistence
    - disable:
        rule: “Some noisy rule”
    - enable:
        rule: “Some interesting rule”
```

Falco

Output Customization

- How to customize your Falco rule output?
- How to add new fields for your JSON object?

Just to give you an idea of the workarounds used atm with the k8s_audit source:

- falco is started with `-p` (since 0.38, before it was awk/sed voodoo on the rule files to append the output of every rule) to add a suffix to every rule output
  `-p " || cluster_name=%jevt.value[/annotations/cluster_name]"`
- this forces falco to add an output_field named `jevt.value[/annotations/cluster_name]` , as this is now used in the output
- as the cluster name is now available as output_field (`jevt.value...`), we can instruct falcosidekick to use it

```
templatedfields:
  cluster_name: '{{ or (index . "jevt.value[/annotations/cluster_name]") "unknown" }}'
loki:
  extralabels: "cluster_name"
```

- use a loki processing pipeline to remove the `|| cluster_name=...` stuff from the alert output/message

Falco

Output Customization

```
append_output:
  - match:
      source: syscall
    extra_output: "on CPU %evt.cpu"
    extra_fields:
      - home_directory: "${HOME}"
      - evt.hostname
```

Falco

Automatic Driver Selection

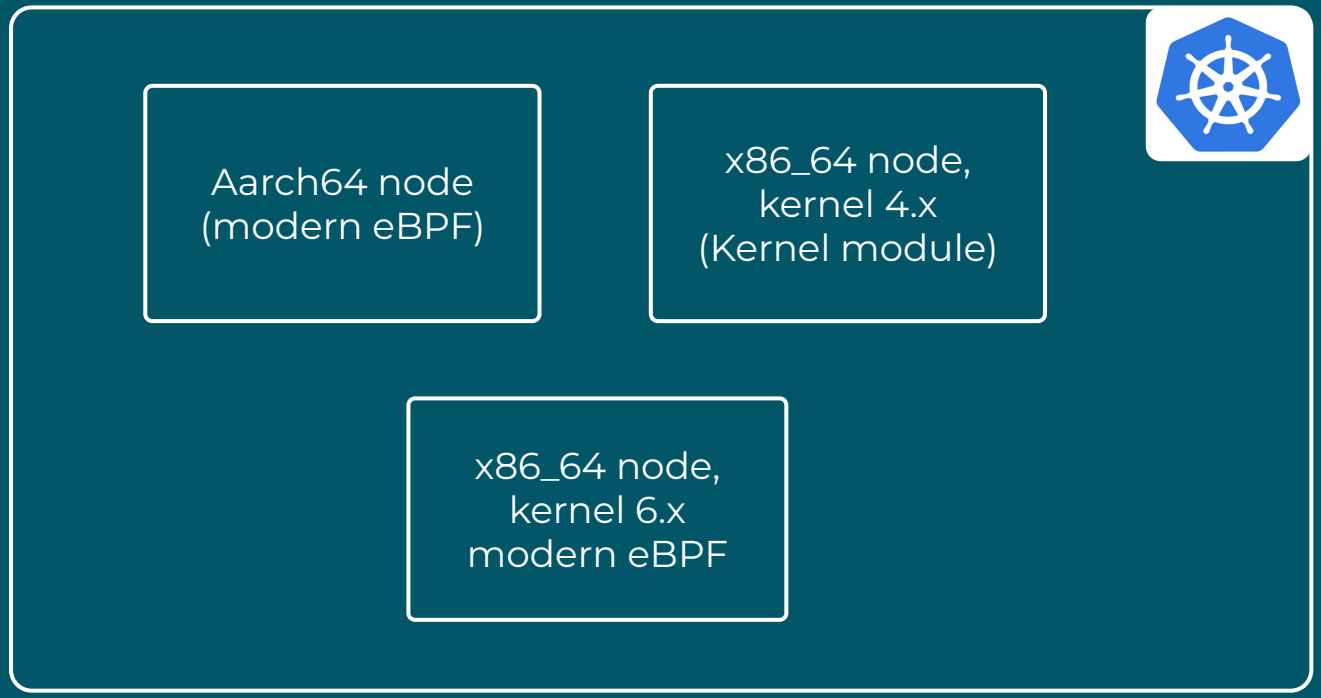
- Real world k8s clusters are complex
- They may have heterogeneous nodes with different architectures, some work with kernel modules, some with eBPF, some with modern eBPF
- **We don’t want to require people to be kernel experts in order to run Falco!**

An operator after attempting a Falco install on a multi-architecture k8s cluster

Falco

Automatic Driver Selection (default since 0.39.0)

```
$ helm install falco falcosecurity/falco --set driver.kind=auto
```

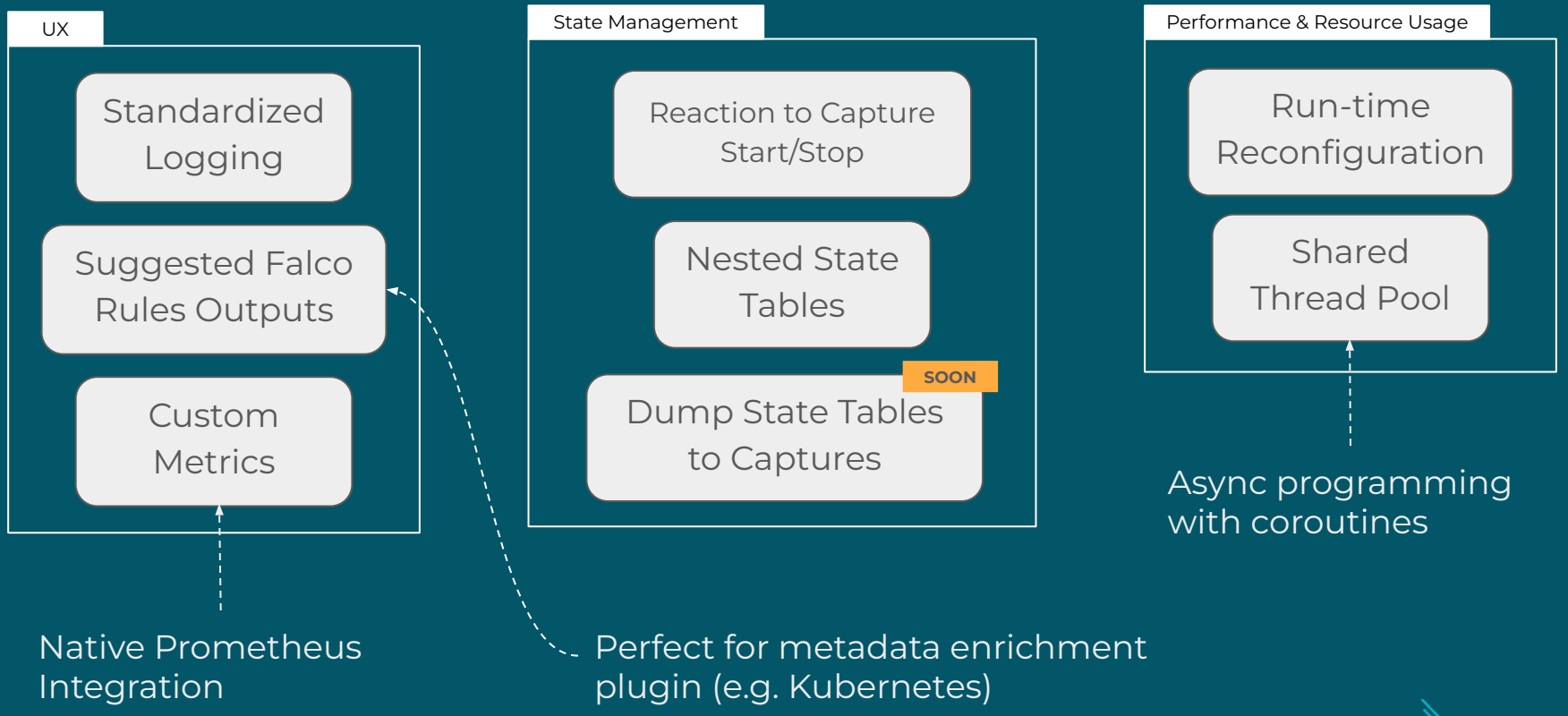


Falco

Falco

Plugin Ecosystem

Falco Plugins: New Framework Features



Falco

Falco Plugins: Community Updates

- **SDK C++** to be released as stable v0.1.0 (next few weeks)
  https://github.com/falcosecurity/plugin-sdk-cpp
- **20M+** installations of **K8S metadata** enrichment plugin
  https://github.com/falcosecurity/plugins/tree/main/plugins/k8smeta
- **Anomaly detection** plugin added to community official catalogue
  https://github.com/falcosecurity/plugins/pull/419
- New **community** plugins: GitLab, Kafka, Keycloak, Journald
  https://github.com/falcosecurity/plugins/blob/main/registry.yaml
- **Modularization** of Falco Core features **container metadata** as a plugin
  https://github.com/falcosecurity/falco/issues/3403#issuecomment-2474112408

Falco

Falco Plugins: Welcoming Rust Developers

- Officially released a **Rust** crate for developing **Falco Plugins**
- **Performance-critical** use cases
- Opportunity for **rejuvenating** and **modularizing** core Falco code
- Supports **all the latest features** of the plugin framework, and can even encode **syscalls events**!
- Memory safe, type-safe, static assertions, boilerplate generation (no downsides, sorry 🤷)

https://github.com/falcosecurity/plugin-sdk-rs
(kudos to our beloved maintainer gnosek)

Falco

Metrics Framework and Prometheus

- New Falco **metrics framework** finalized and **officially released**
- Integrated **Prometheus** Exporter
  - falco-exporter to be deprecated
  - Optional periodic snapshot also as Falco alerts
- Insights about the **state of the node** (Falco sees everything 📷)
  - CPU & mem usage, dropped events, security alerts, node's processes...
- We use it ourselves in our Prow cluster
- Native support in **Falco plugins**

https://monitoring.prow.falco.org/d/ddwe2ug4nfi0wb/falco?orgId=1

Falco

In the Works: Multi-Value Field Transformers

- Aiming to make our **security rules language even more powerful**
- Goal is to achieve fully-fledged function-like multi-arg operators
- Usable both in Falco rules **conditions** and **outputs**
- We expect new use cases to **grow dramatically**
- In the works, stay tuned...

```
rule: Process deleting its own executable
desc: [...]
condition: >
   evt.type = unlink and
   proc.exepath = val(fs.path.name)
```

```
getopt(proc.cmdline, '-d') = "foo"
join(proc.aexepath, ',') contains "runc"
div(proc.fdopencount, proc.fdlimit) >= 0.1
...
```

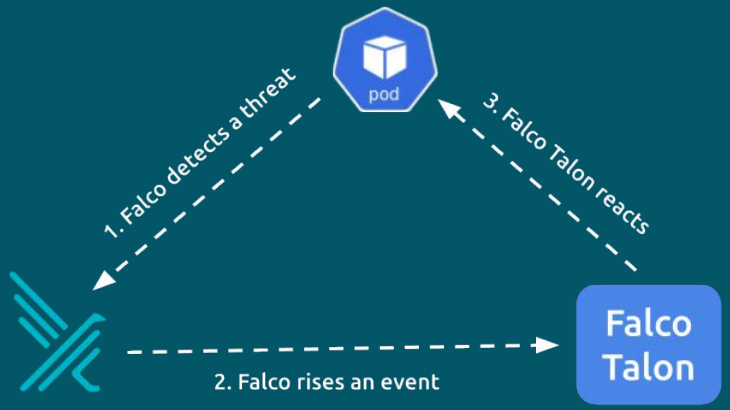
Falco

Falco

Ecosystem Projects

Falco Talon

- Ecosystem Project that was donated to Falco (thanks to Thomas Labarussias and all contributors)
- Response Engine for managing threats in Kubernetes clusters
- It enhances the solutions proposed by the Falco community with a no-code tailor-made solution.
- With easy rules, you can react to events from Falco in milliseconds.



Falco

Example of Falco Talon Action/Rule

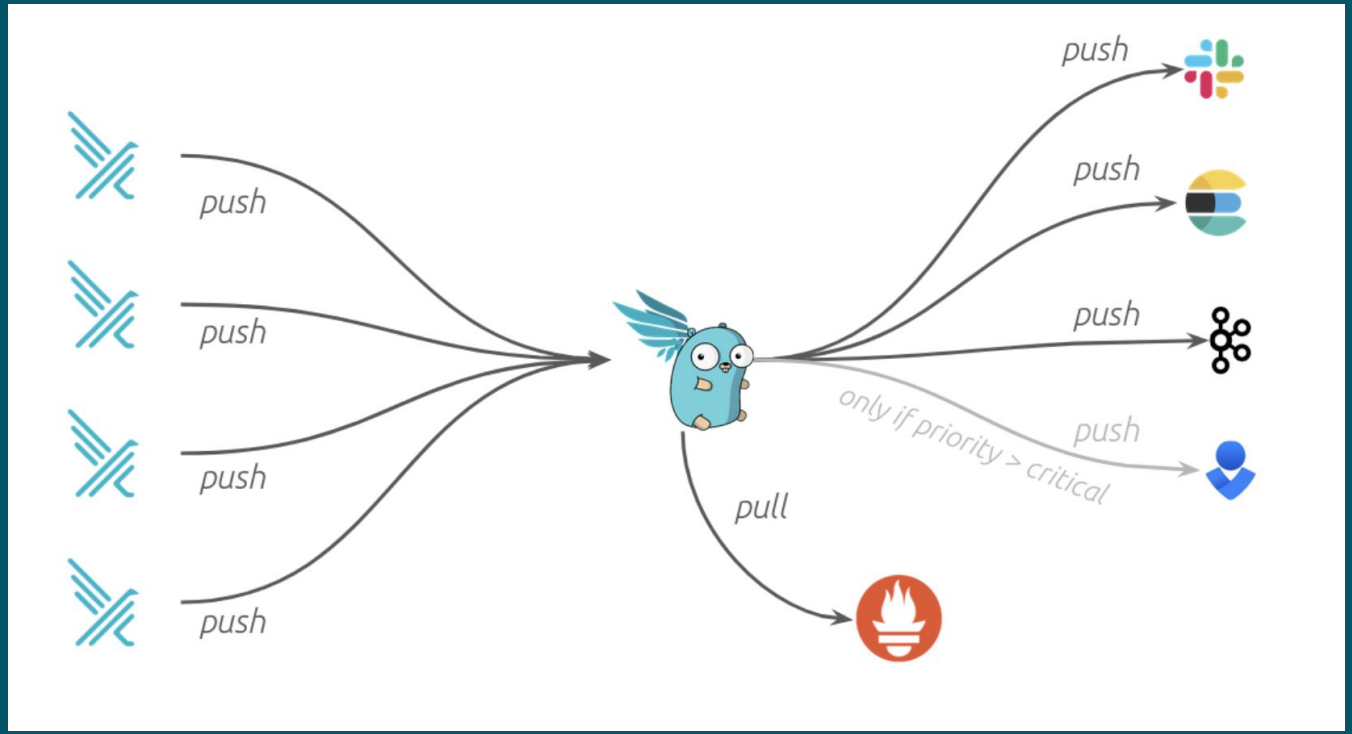
```yaml
- action: Terminate Pod
  description: terminate the pod if it doesn't belong to a statefulset
  actionner: kubernetes:terminate
  parameters:
    ignoreDaemonsets: false
    ignoreStatefulsets: true

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

- rule: Suspicious outbound connection
  description: Block suspicious outbound connections and terminate the pod
  match:
    rules:
      - Unexpected outbound connection destination
  actions:
    - action: Get last logs
      actionner: kubernetes:log
      parameters:
        tail_lines: 10
      output:
        target: aws:s3
        parameters:
          bucket: my-bucket
          prefix: /logs/
    - action: Disable outbound connections
      ignore_errors: true
    - action: Terminate Pod # ref to a re-usable action
      parameters:
        gracePeriods: 2
```

Falcosidekick

- It takes a Falco events and forward them to different outputs in a fan-out way.



Falco

Join our community

#falco channel on the
 Kubernetes Slack 👨‍💻

Falco **community call**
every Wednesday 🤝

📨 Mailing list 📨
cncf-falco-dev@lists.cncf.io

👉 falco.org 👈

Falco

Falco

falco.org

Thank You!

We'd love to hear your thoughts 👇

Follow us right now 👇

Graduated Projects Celebration 🎉

4:00pm - 4:30pm
Project Pavilion
Level 1 - Halls A-C + 1-5