Falco Deep Dive: Extending Event Sources for Kubernetes Audit

Kaizhe Huang, Sysdig

About me.

Kaizhe Huang

Security Researcher, Sysdig


@kaizhe


@kaizhe

Join the community.

Website

- https://falco.org

Public Slack

- http://slack.sysdig.com/
- https://sysdig.slack.com/messages/falco

Blog

- https://sysdig.com/blog/tag/falco/

Workshop

- https://github.com/falcosecurity/falco-security-workshop

Github

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

Documentation

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

Docker Hub

- https://hub.docker.com/r/falcosecurity/falco/

@mfdii

Falco Team

@leodido

Leonardo

@fntlnz

Lorenzo

@mstemm

Mark

@ldegio

Loris

@mfdii

Ducy

Help Wanted

- **Profiles Library**
  - Expand predefined rule sets for common use cases and applications
- **Prometheus Metrics Exporter**
  - Expose metric counts for rules fired, etc
- **GRPC Based Outputs**
  - Streaming based outputs for alerts
- **Helm Chart**
  - General maintenance
- **Kubernetes Operator**
  - Add features to improve operator functionality, developer defined rules, etc
- **New Event Sources**
  - Container runtime events, application level events, more

@mfdii

Falco.

A behavioral activity monitor

- Detects suspicious activity defined by a set of rules
- Uses sysdig's flexible and powerful filtering expressions

With full support for containers/orchestration

- Utilizes sysdig's container & orchestrator support

And flexible notification methods

- Alert to files, standard output, syslog, programs

Open Source

- Anyone can contribute rules or improvements

@mfdii

Falco: a CNCF sandbox project.

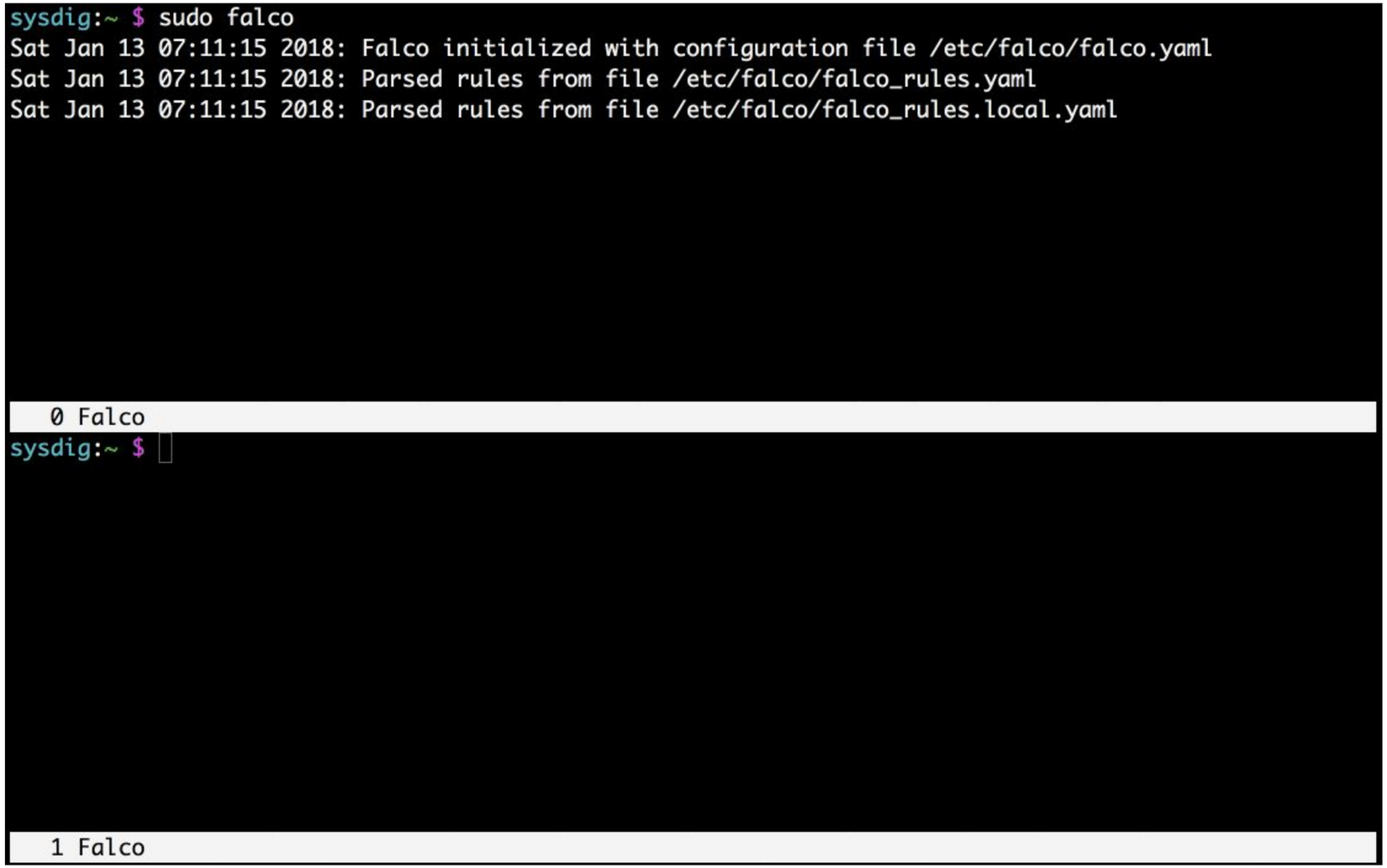
Runtime security for cloud native platforms

- Detect abnormal behavior in applications, containers, and hosts.
- Audit orchestrator activity.

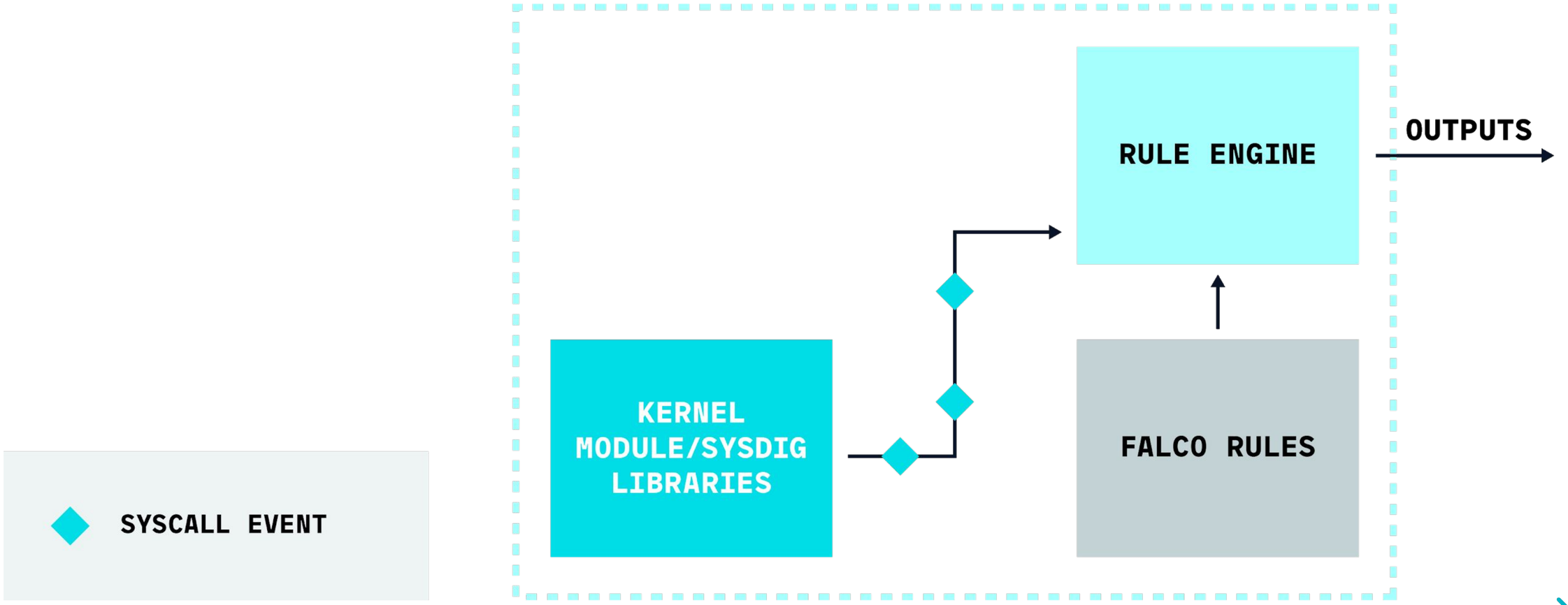
Cloud Native Computing Foundation (CNCF)

- Sandbox level project
- sysdig.com/blog/falco-cncf-sandbox

Example.


sysdig:~ $ sudo falco
Sat Jan 13 07:11:15 2018: Falco initialized with configuration file /etc/falco/falco.yaml
Sat Jan 13 07:11:15 2018: Parsed rules from file /etc/falco/falco_rules.yaml
Sat Jan 13 07:11:15 2018: Parsed rules from file /etc/falco/falco_rules.local.yaml
0 Falco
sysdig:~ $
1 Falco

Falco internal architecture.



Additional event sources.

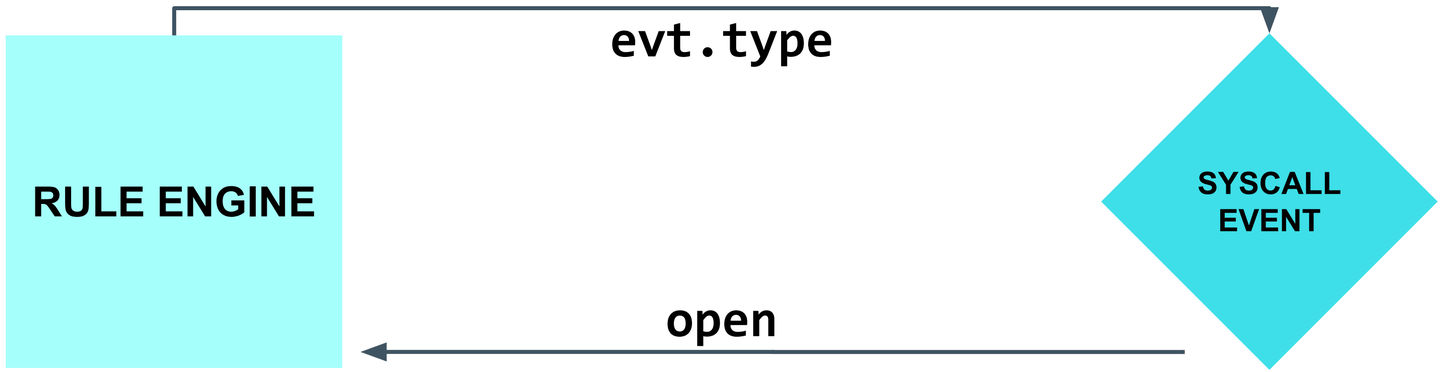
Falco extracts values from events using *filter fields*

```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```

Additional event sources.

Falco extracts values from events using *filter fields*

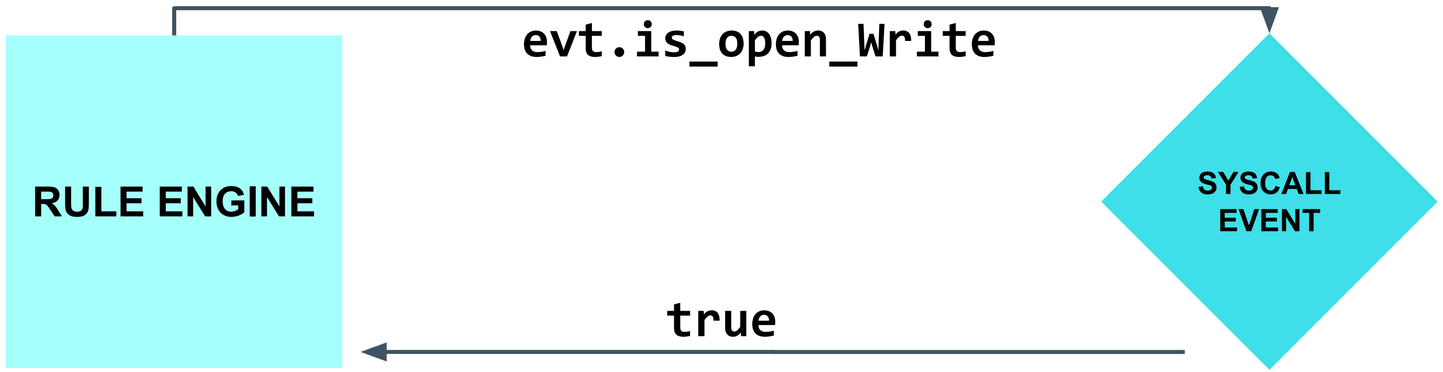
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```



Additional event sources.

Falco extracts values from events using *filter fields*

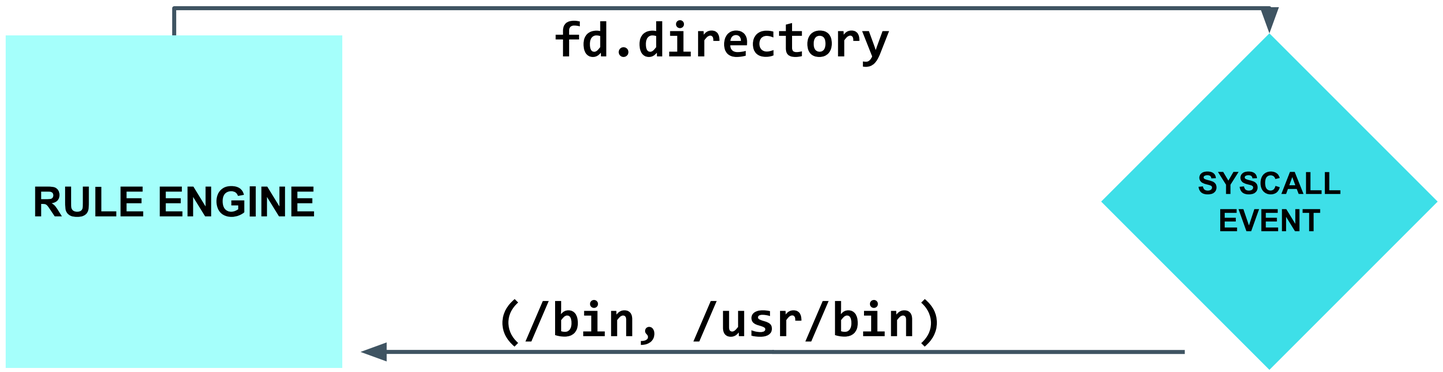
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```



Additional event sources.

Falco extracts values from events using *filter fields*

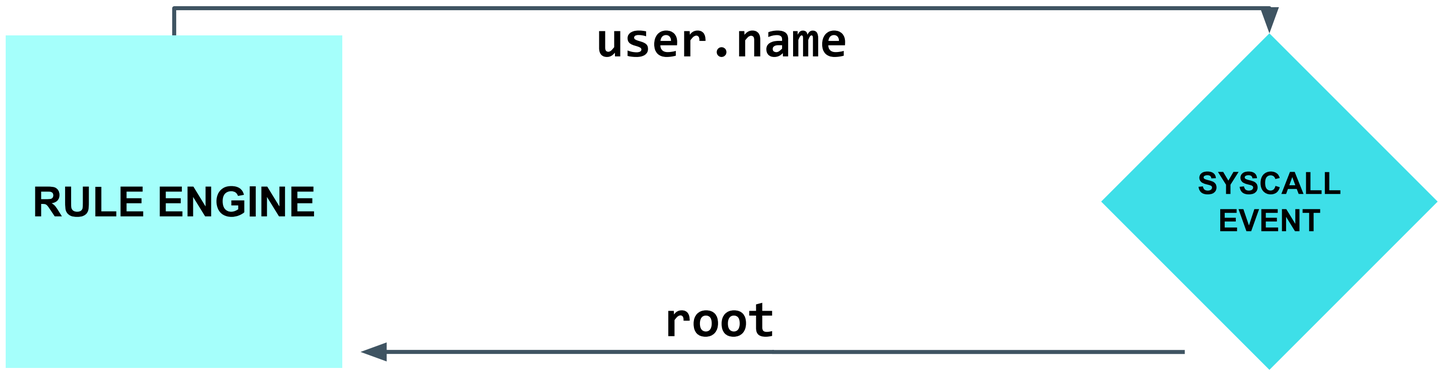
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```



Additional event sources.

Falco extracts values from events using *filter fields*

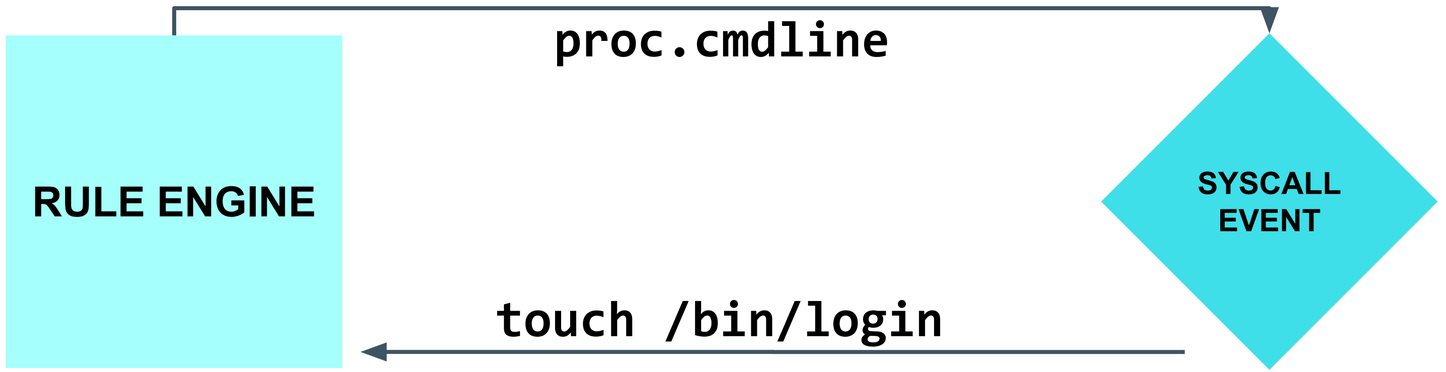
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```



Additional event sources.

Falco extracts values from events using *filter fields*

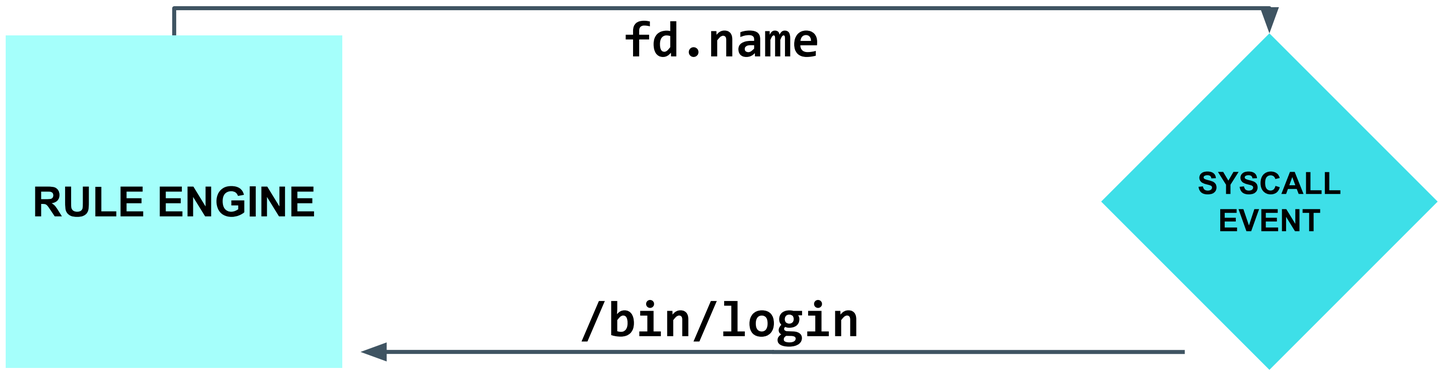
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```



Additional event sources.

Falco extracts values from events using *filter fields*

```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```



Additional event sources.

```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and and evt.is_open_write=true and fd.directory in
(/bin, /usr/bin)
  output: >
        File below a known binary directory opened for writing (user=%user.name
        command=%proc.cmdline file=%fd.name)
  priority: WARNING
```

Falco can be extended to work on other “kinds” of events

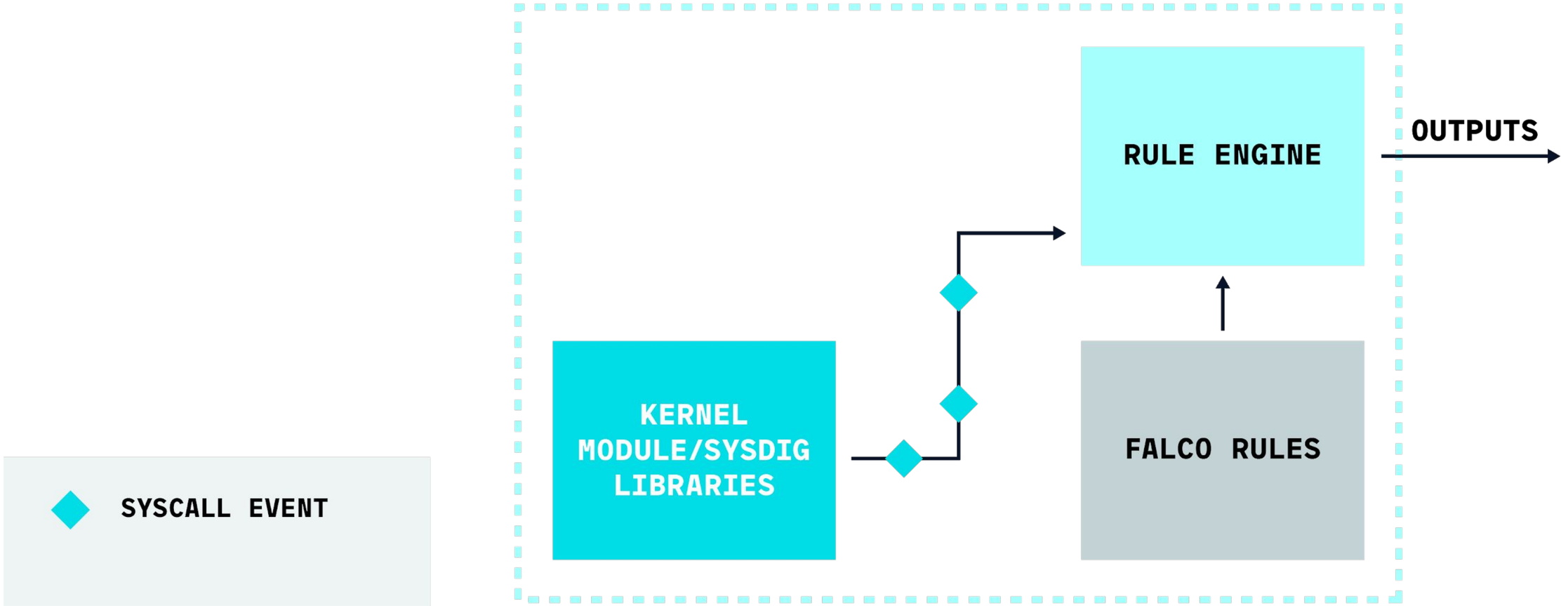
Kubernetes audit events.

- New in K8s v1.11
- Provides chronological set of records documenting changes to cluster
- Each record is a JSON object
- Audit policy controls which events are included in event log
- Log backend controls where events are sent
  - Log file
  - Webhook
  - AuditSink (alpha as of 1.13)

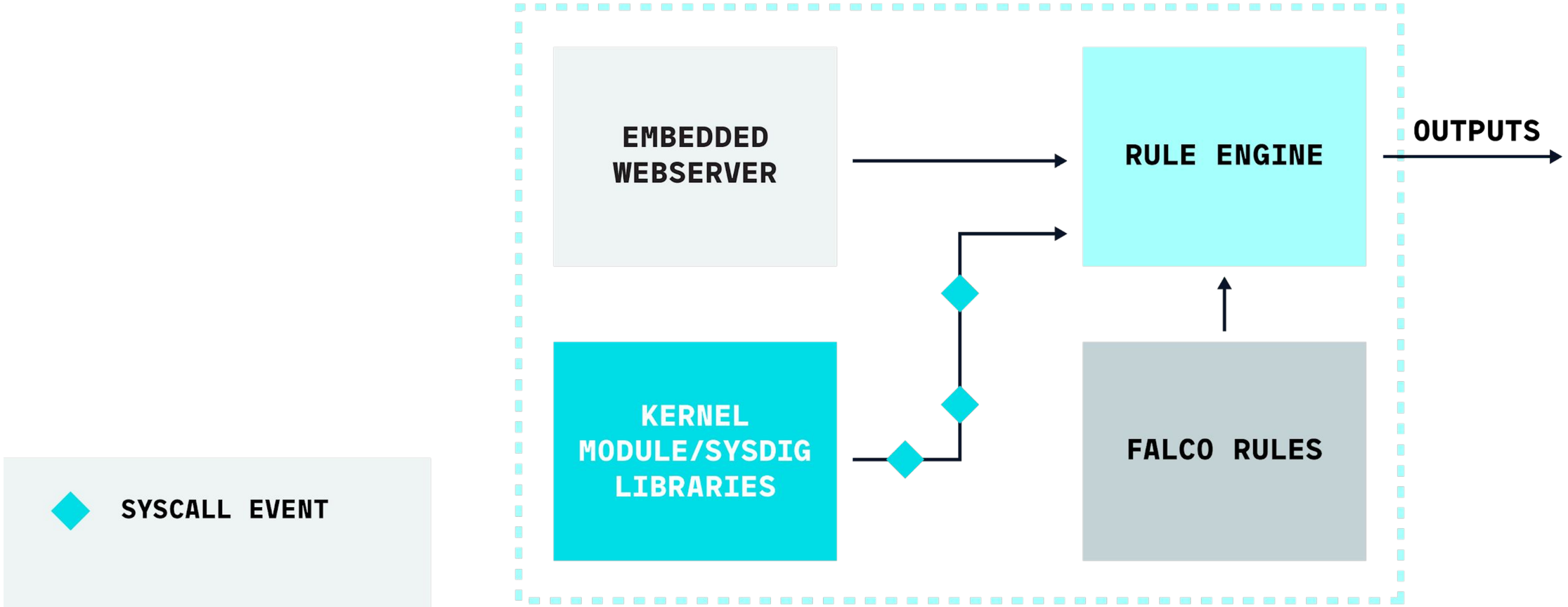
K8s audit events.

```
{
    "kind": "Event",
    "timestamp": "2018-10-26T13:00:25Z",
    "stage": "ResponseComplete",
    "verb": "delete",
    "requestURI": "/api/v1/namespaces/foo",
    "user": { "username": "minikube-user" },
    "responseStatus": { "code": 200 },
    "objectRef": { "resource": “namespaces", "namespace": “foo” },
    "level": "Request",
    "auditID": "693f4726-2430-450a-83e1-123c050fde98",
    "annotations": { "authorization.k8s.io/decision": "allow" }
}
```

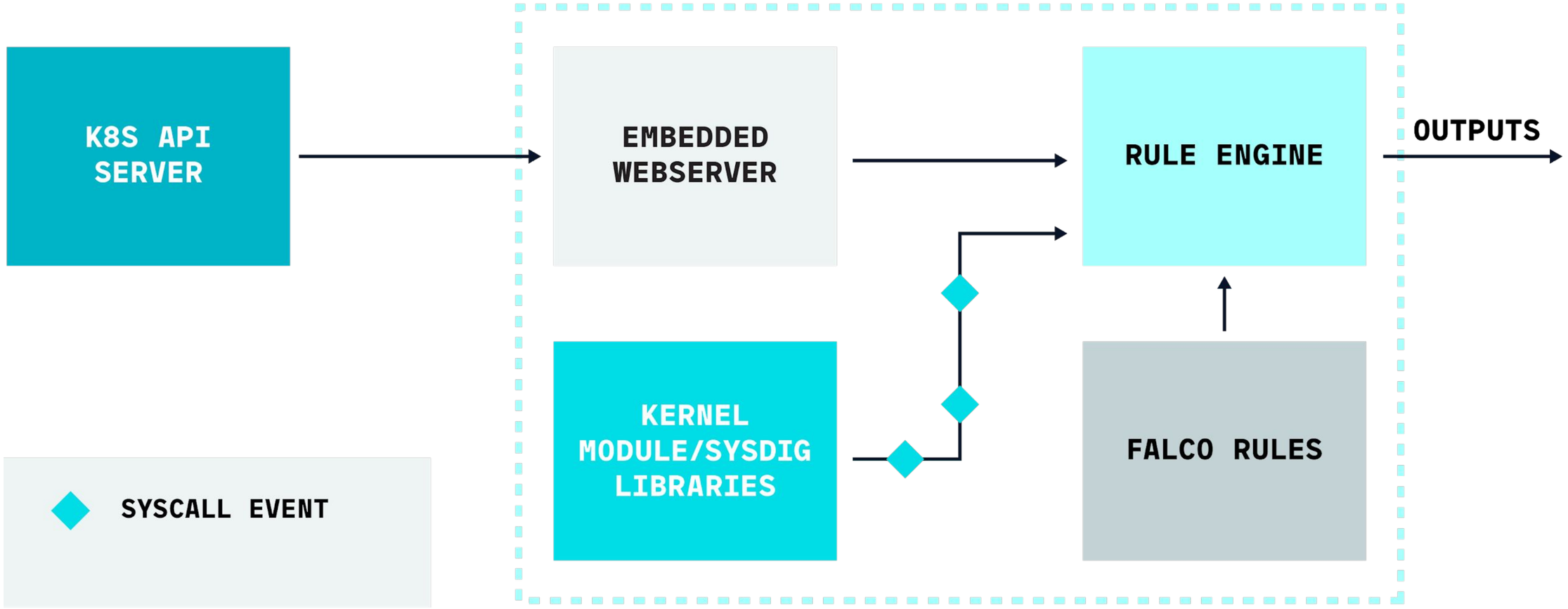

Falco internal architecture.



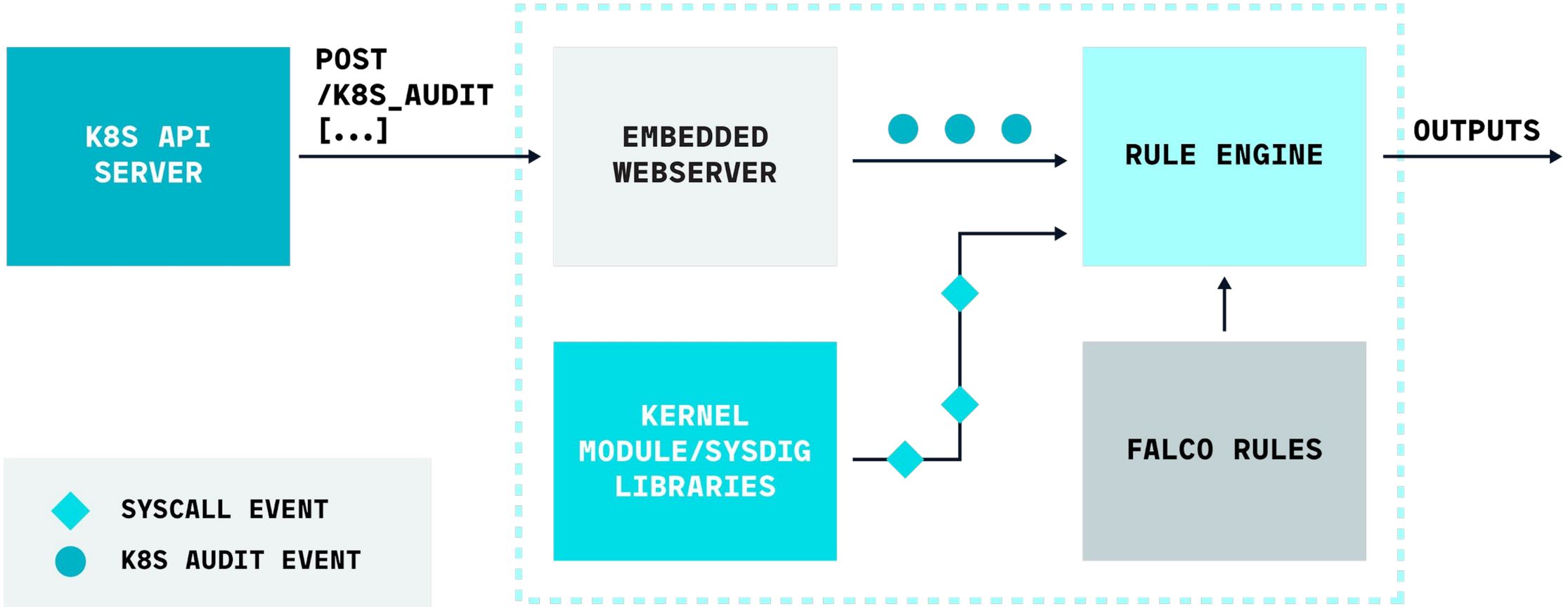
Falco internal architecture.



Falco internal architecture.



Falco internal architecture.



Supporting K8s audit events.

Create a new “Generic Event” interface

- Event time, ability to extract values using fields

Create a K8s Audit Event object

- Event data is json object, stored in event

Define new fields to extract values from K8s Audit Events

- Uses *Json Pointers* to extract values

Each Falco Rule now has a source

- Default “syscall”, “k8s_audit” for K8s Audit Events

JSON pointers.

```
{
    "foo": 1,
    "bar": {
        "baz": "tee"
    },
    "arr": [1, 3, 4]
}
```

- “/foo” -> 1
- “/bar/baz” -> “tee”
- “/arr/0” -> 1
- “/arr/2” -> 4

Accessing K8s audit events.

```
{
    "verb": "delete",
    "responseStatus": { "code": 200 },
    "objectRef": {
        "resource": “namespaces",
        "namespace": “foo”
    },
    "level": "Request"
}
```

- “/verb” -> “delete”
- “/responseStatus/code” -> 200
- “/objectRef/resource” -> “namespaces”

K8s audit event fields.

`jevt.value[<json_pointer>]`

- Access any field from json object

`jevt.time`

- Access event timestamp

`ka.verb, ka.uri, ka.user.name, ka.target.resource, …`

- Access specific values from object
- Implemented as macros:
  - `ka.verb -> jevt.value[/verb]`
  - `ka.target.resource -> jevt.value[/objectRef/resource]`
- Full list: falco —list=k8s_audit

K8s audit rule example.

```
- macro: contains_private_credentials
  condition: >
    (ka.req.configmap.obj contains "aws_access_key_id" or
     ka.req.configmap.obj contains "aws_s3_access_key_id" or
     ka.req.configmap.obj contains “password")

- macro: configmap
  condition: ka.target.resource=configmaps

- macro: modify
  condition: (ka.verb in (create,update,patch))

- rule: Create/Modify Configmap With Private Credentials
  desc: Detect creating/modifying a configmap containing a private
credential
    (aws key, password, etc.)
  condition: configmap and modify and contains_private_credentials
  output: K8s configmap with private credential (user=%ka.user.name
          verb=%ka.verb name=%ka.req.configmap.name
          configmap=%ka.req.configmap.name config=%ka.req.configmap.obj)
  priority: WARNING
  source: k8s_audit
  tags: [k8s]
```


Demo.

Join the community.

Website

- https://falco.org

Public Slack

- http://slack.sysdig.com/
- https://sysdig.slack.com/messages/falco

Blog

- https://sysdig.com/blog/tag/falco/

Workshop

- https://github.com/falcosecurity/falco-security-workshop

Github

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

Documentation

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

Docker Hub

- https://hub.docker.com/r/falcosecurity/falco/

@mfdii

Thank You!