



Detecting K8s Audit Events with Falco.

Mark Stemm, Falco Engineer

What is 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

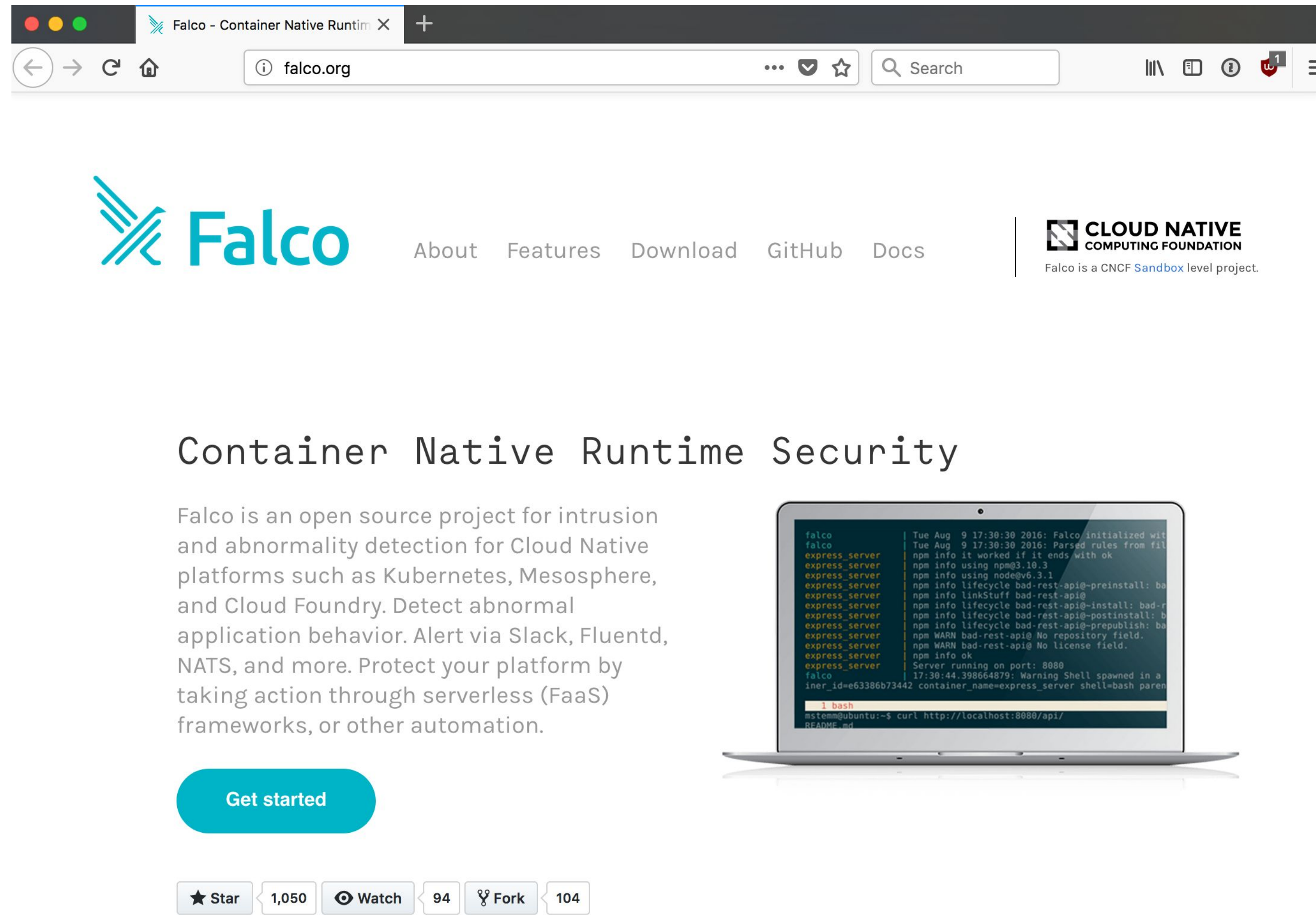
- Alert to files, standard output, syslog, programs

Open Source

- Anyone can contribute rules or improvements



Falco joins the CNCF Sandbox.



The screenshot shows a browser window with the URL `falco.org`. The page features the Falco logo, navigation links for 'About', 'Features', 'Download', 'GitHub', and 'Docs', and the Cloud Native Computing Foundation logo with the text 'Falco is a CNCF Sandbox level project.' The main heading is 'Container Native Runtime Security'. Below it, a paragraph describes Falco as an open source project for intrusion and abnormality detection for Cloud Native platforms like Kubernetes, Mesosphere, and Cloud Foundry. A 'Get started' button is present. At the bottom, GitHub statistics show 1,050 stars, 94 forks, and 104 watchers.

Falco - Container Native Runtime X +

falco.org

Falco About Features Download GitHub Docs

CLOUD NATIVE COMPUTING FOUNDATION
Falco is a CNCF [Sandbox](#) level project.

Container Native Runtime Security

Falco is an open source project for intrusion and abnormality detection for Cloud Native platforms such as Kubernetes, Mesosphere, and Cloud Foundry. Detect abnormal application behavior. Alert via Slack, Fluentd, NATS, and more. Protect your platform by taking action through serverless (FaaS) frameworks, or other automation.

[Get started](#)

★ Star 1,050 👁 Watch 94 🍴 Fork 104



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
```

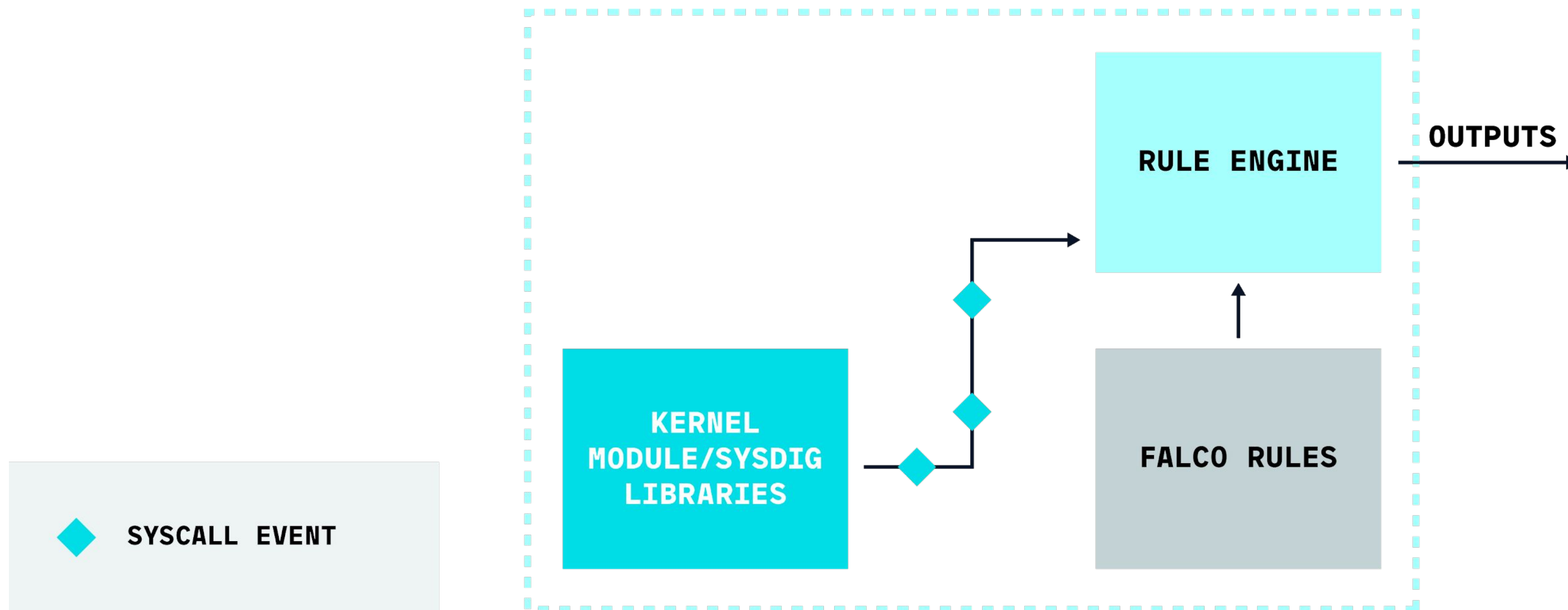
```
0 Falco
```

```
sysdig:~ $
```

```
1 Falco
```



Falco internal architecture.



Additional event sources.

Falco extracts values from events using *filter fields*

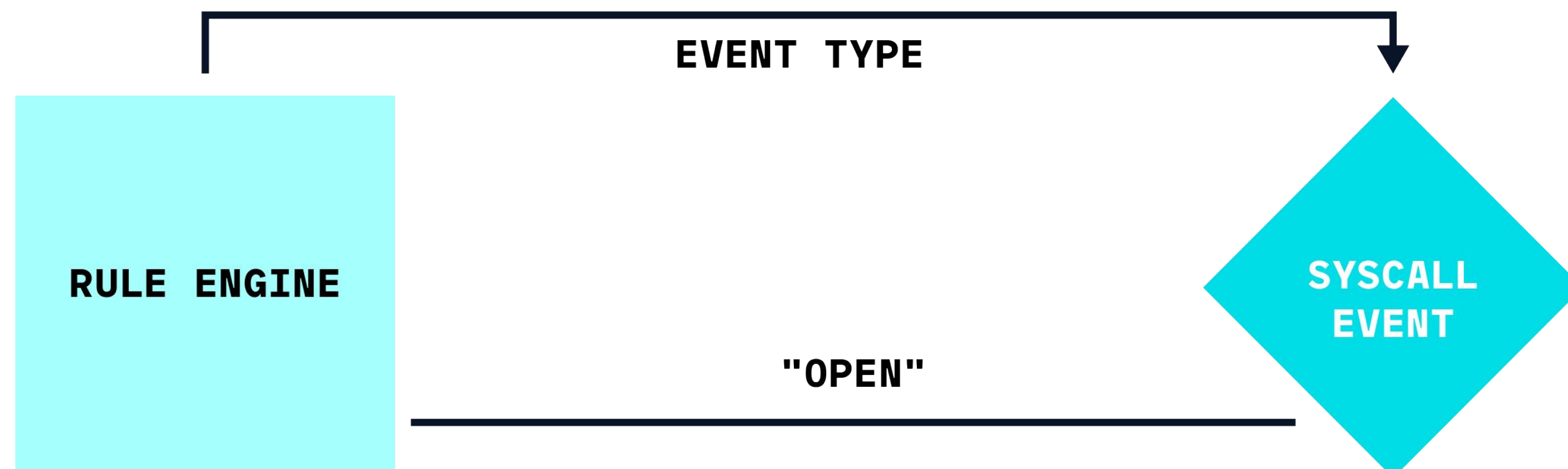
```
- rule: Write Below Binary Dir
desc: An attempt to write to any file below a set of binary directories
condition: evt.type=open and fd.directory in (/bin, /usr/bin)
output: >
    File below a known binary directory opened for writing (user=%user.name
    command=%proc.cmdline file=%fd.name)
priority: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



Additional event sources.

Falco extracts values from events using *filter fields*

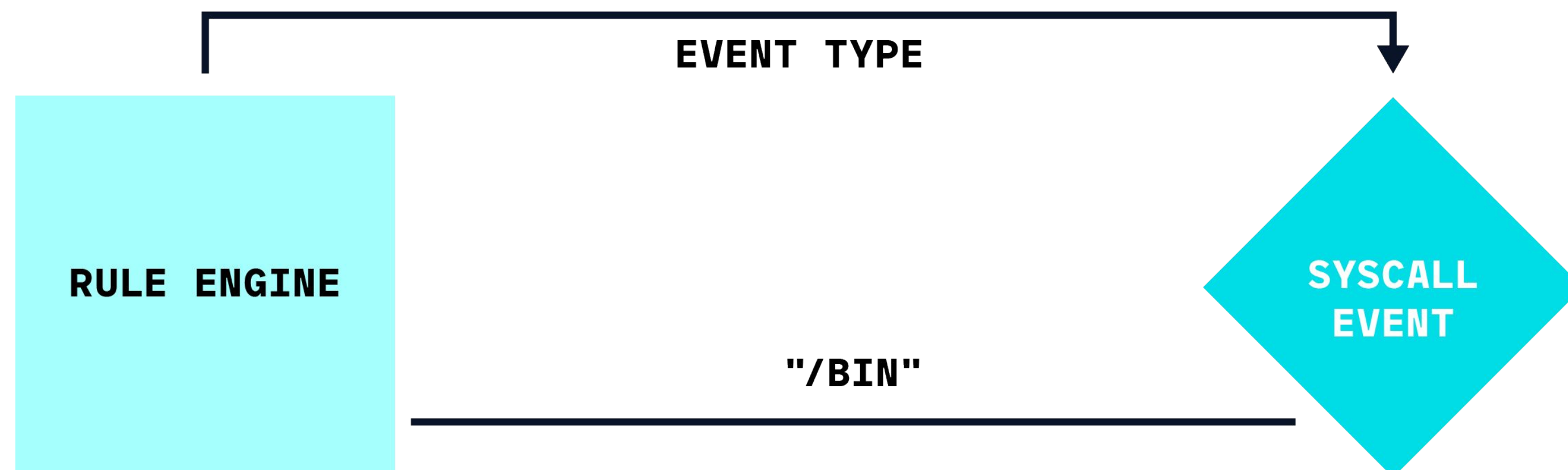
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and fd.directory in (/bin, /usr/bin)
  output: >
           File below a known binary directory opened for writing (user=%user.name
           command=%proc.cmdline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



Additional event sources.

Falco extracts values from events using *filter fields*

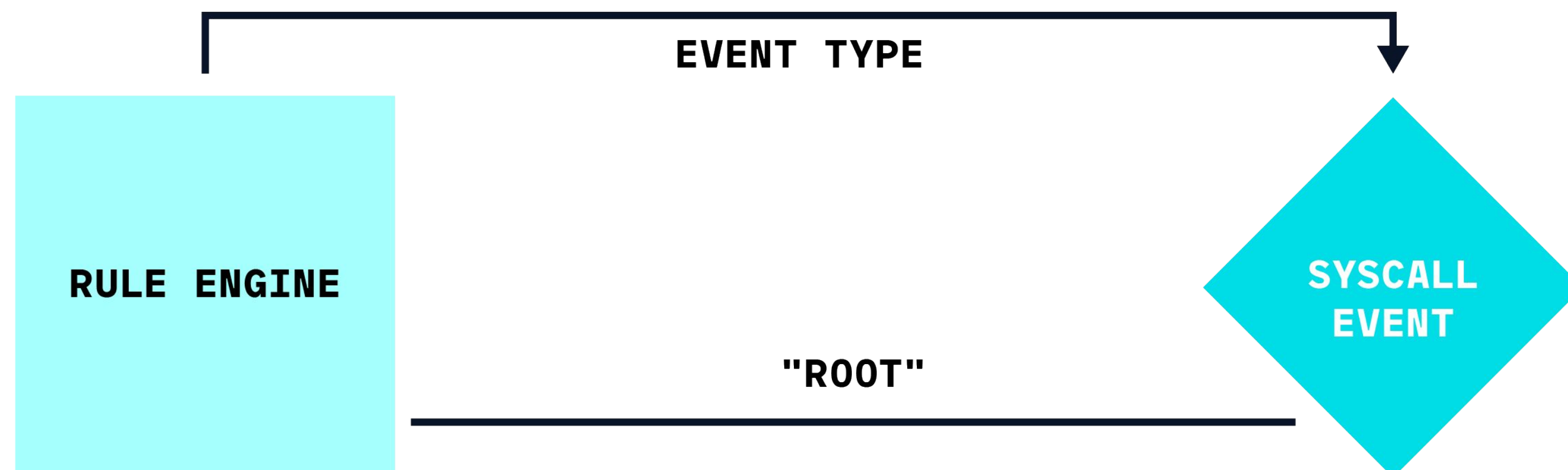
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and fd.directory in (/bin, /usr/bin)
  output: >
    File below a known binary directory opened for writing (user=%user.name
    command=%proc.cmdline file=%fd.name)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

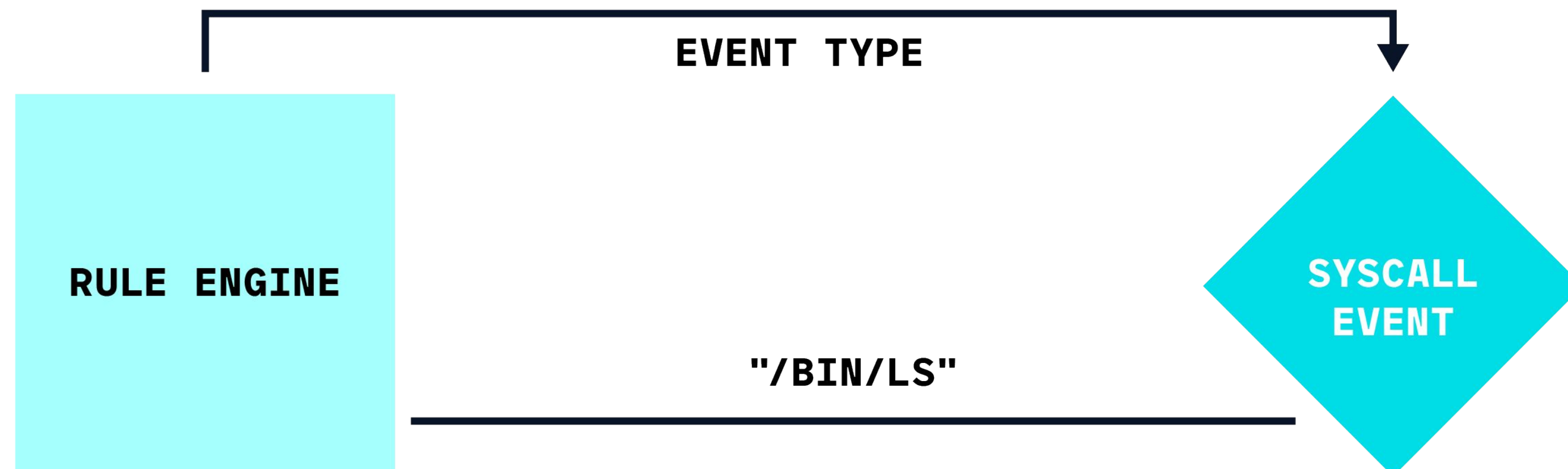
```
- rule: Write Below Binary Dir
  desc: An attempt to write to any file below a set of binary directories
  condition: evt.type=open and fd.directory in (/bin, /usr/bin)
  output: >
    File below a known binary directory opened for writing (user=%user.name
    command=%proc.cmdline file=%fd.name)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



K8s 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

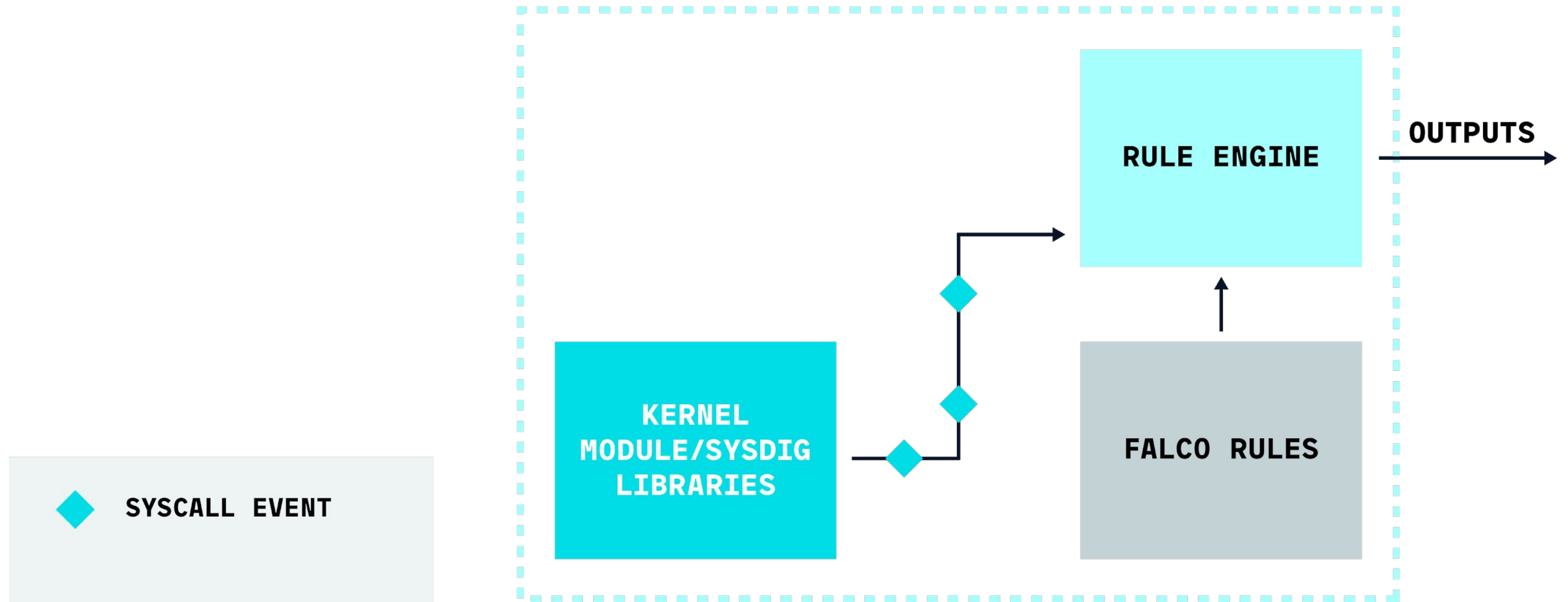


K8s audit events.

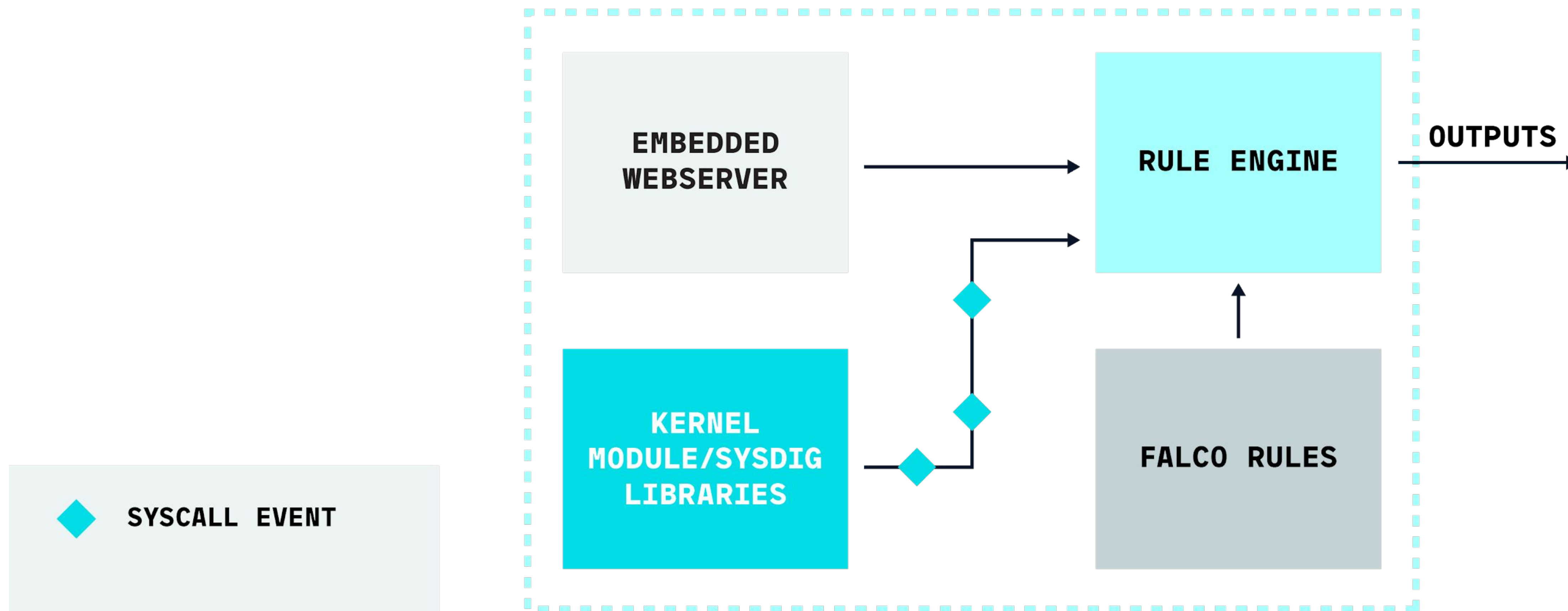
```
{
  "kind": "Event",
  "timestamp": "2018-10-26T13:00:25Z",
  "stage": "ResponseComplete",
  "verb": "delete",
  "requestURI": "/api/v1/namespaces/foo",
  "user": { "username": "minikube-user" },
  "responseStatus": { "code": 200 },
  "objectRef": { "resource": "namespaces", "namespace": "foo" },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



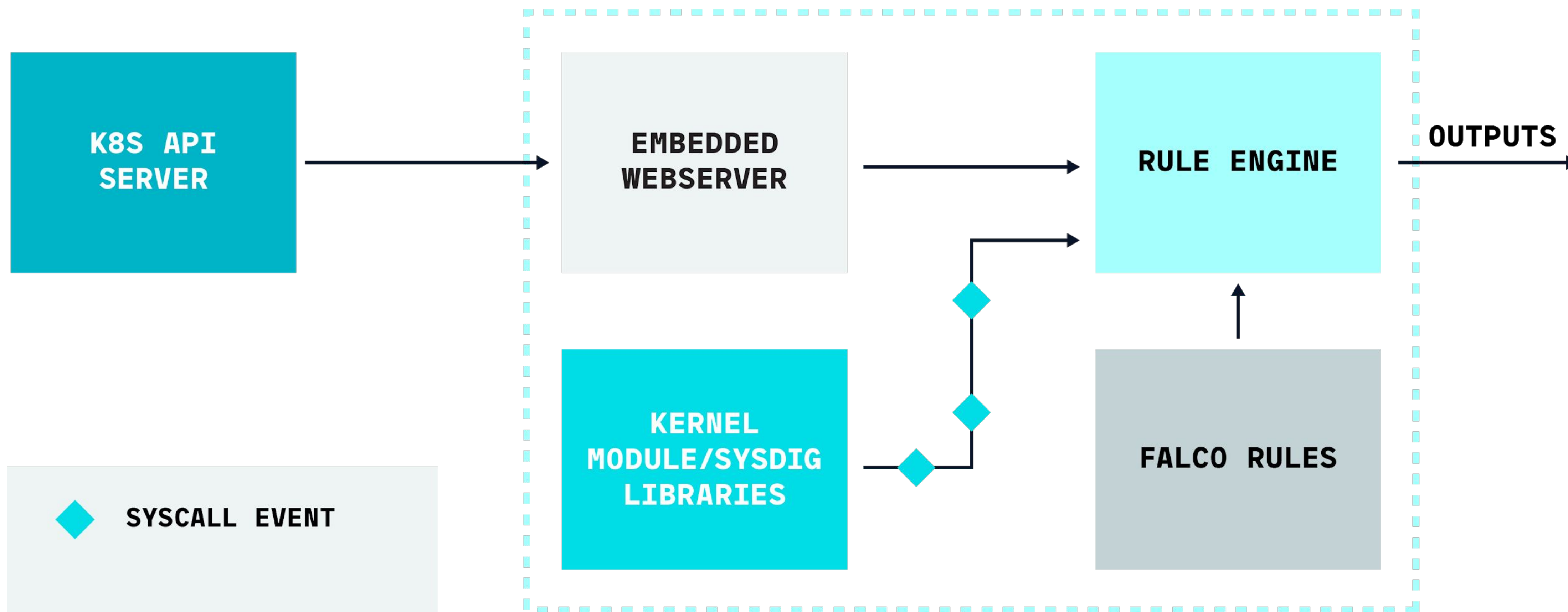
Falco internal architecture.



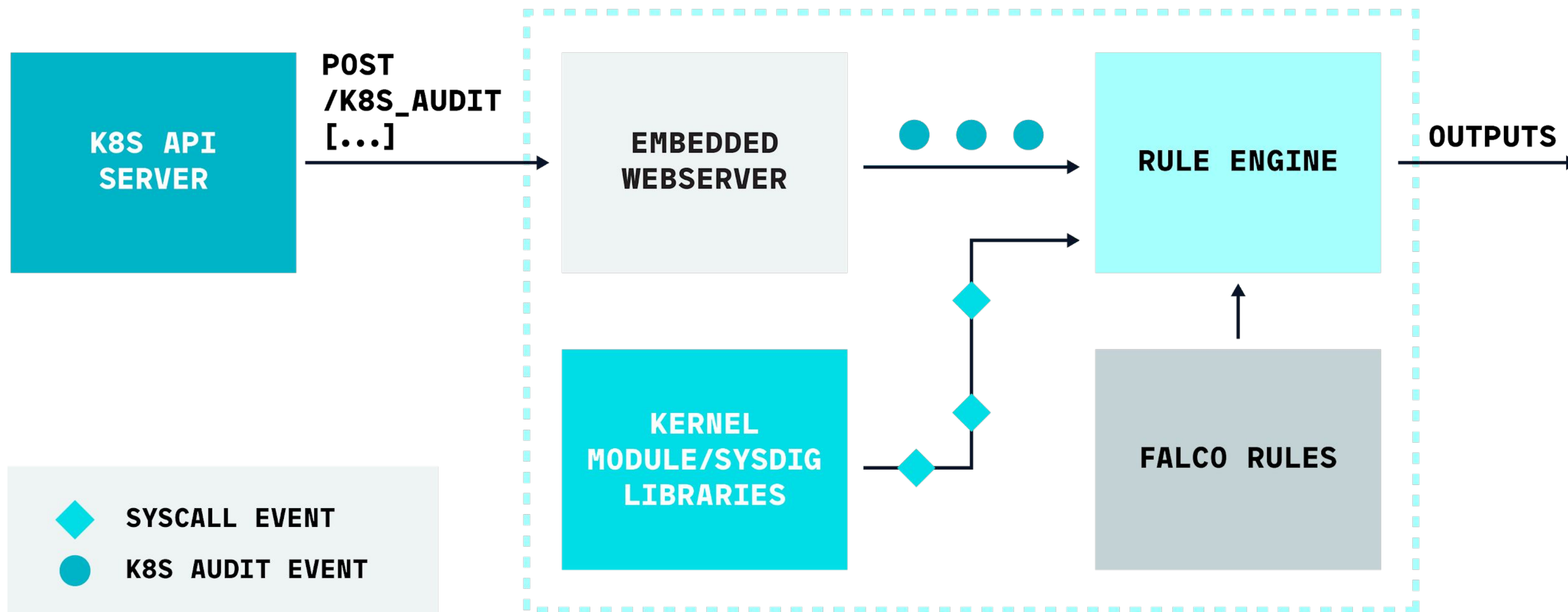
Falco internal architecture.



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



- **CNCF Home of Falco**
 - <http://falco.org>
- **Public Slack**
 - <https://sysdig.slack.com/messages/falco>
- **Blog**
 - <https://bit.ly/FalcoBlogPosts>
- **Sysdig Secure**
 - <http://sysdig.com/product/secure>



Learn more.

- **K8s Audit Support**
 - <https://github.com/falcosecurity/falco/wiki/K8s-Audit-Event-Support>
- **Github**
 - <https://github.com/falcosecurity/falco>
 - Pull Requests welcome!
- **Wiki**
 - <https://github.com/falcosecurity/falco/wiki>
- **Docker Hub**
 - <https://hub.docker.com/r/falcosecurity/falco/>



Thank You!

