OpenShift - Splunk

Melbourne Splunk Meetup

Brendan Wreford - Michiel Kalkman

Goal

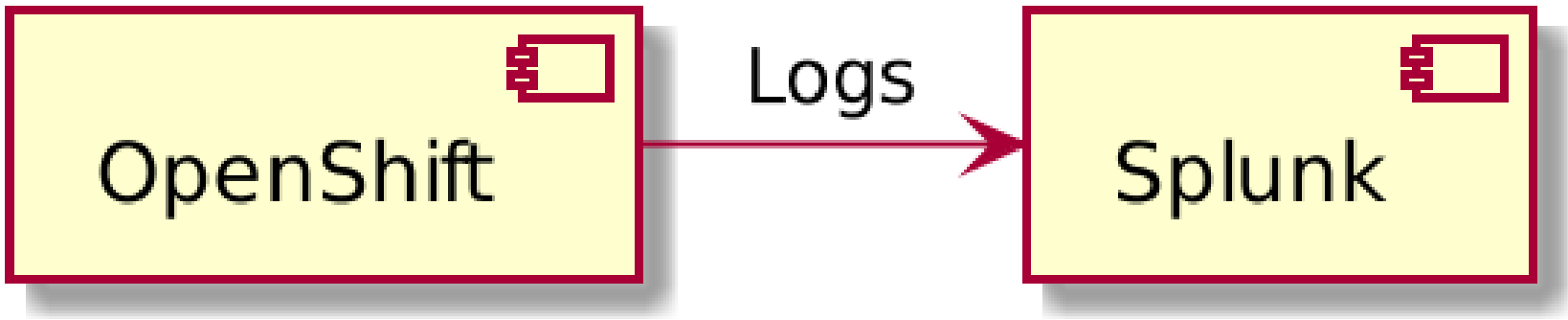

Figure 1: All OpenShift log events to Splunk

OpenShift

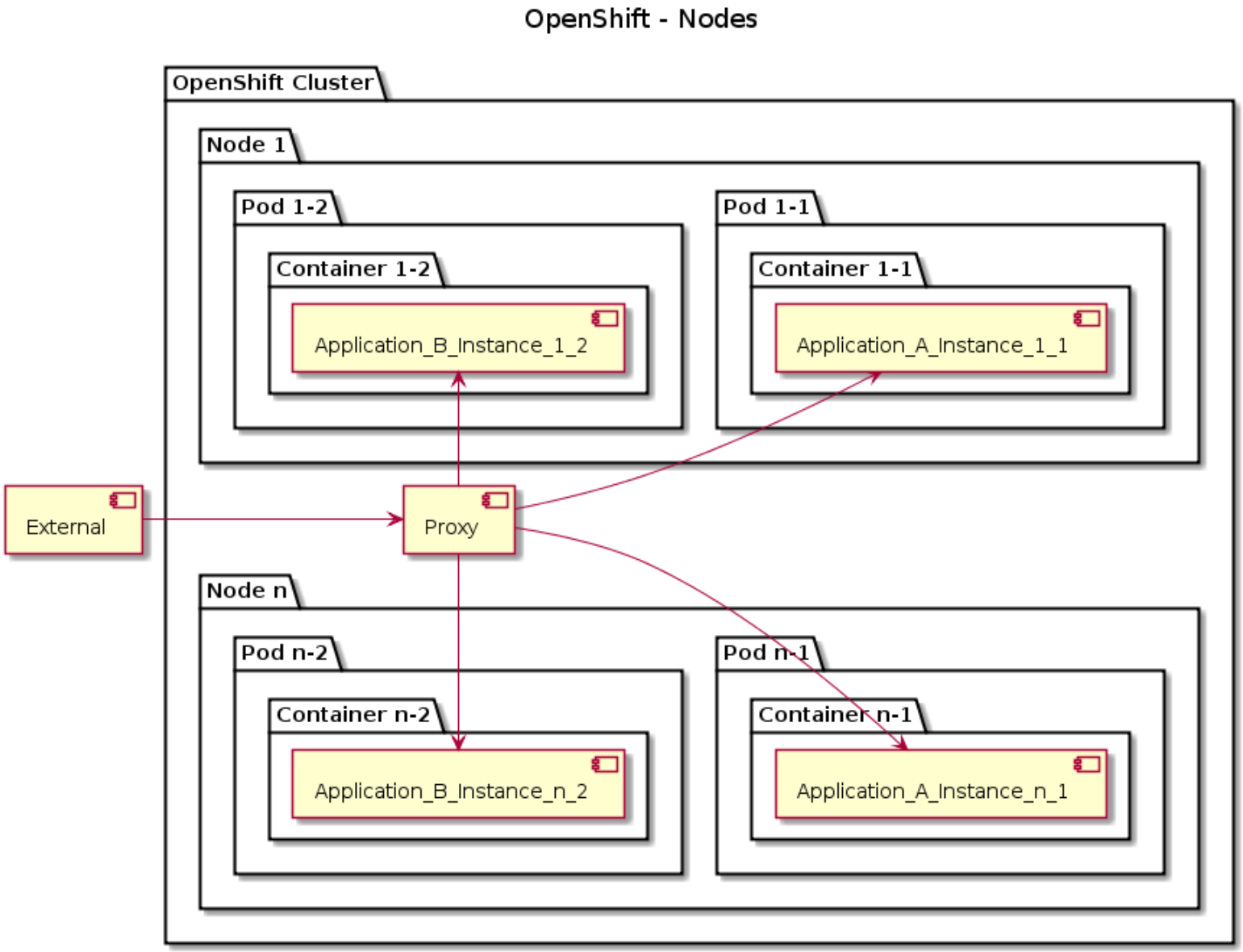

Figure 2: Nodes, Pods

OpenShift Pods - DaemonSet

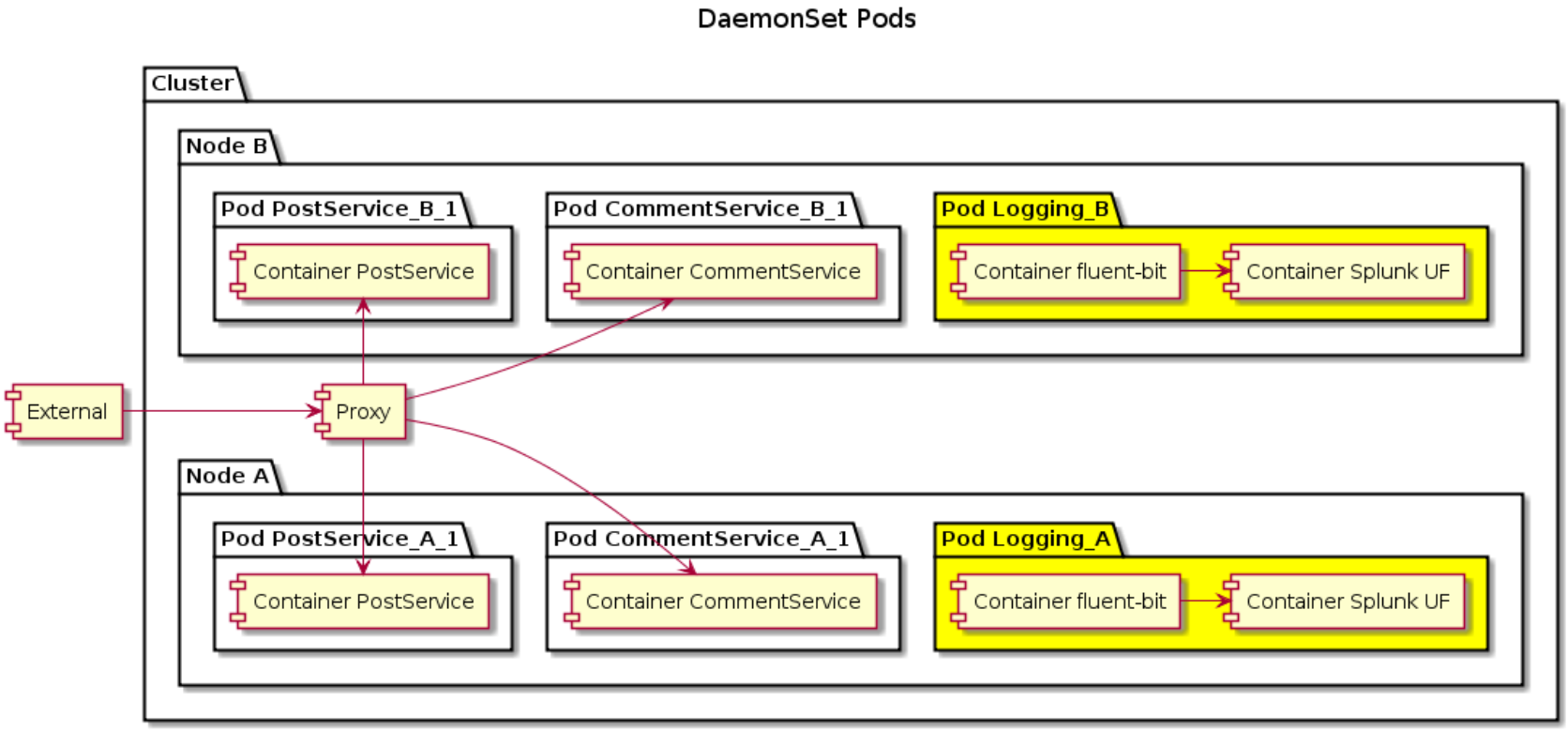

Figure 3: DaemonSet Pods

12-factor apps - Factors

1. One codebase, many deploys
2. Explicitly declare and isolate dependencies
3. Store config in the environment
4. Backing services are attached resources
5. Separate build, release, run stages
6. Execute the app as one or more stateless processes
7. Export services via port binding
8. Scale out via the process model
9. Robustness with fast startup / graceful shutdown
10. Dev/prod parity
11. **Logs are streams**
12. Admin tasks are one-off processes

`https://12factor.net/`

12-factor apps - Logs

> *A twelve-factor app never concerns itself with routing or storage of its output stream. It should not attempt to write to or manage logfiles. Instead,* ***each running process writes its event stream, unbuffered, to stdout.***

Source - `https://12factor.net/logs`

Solution - components

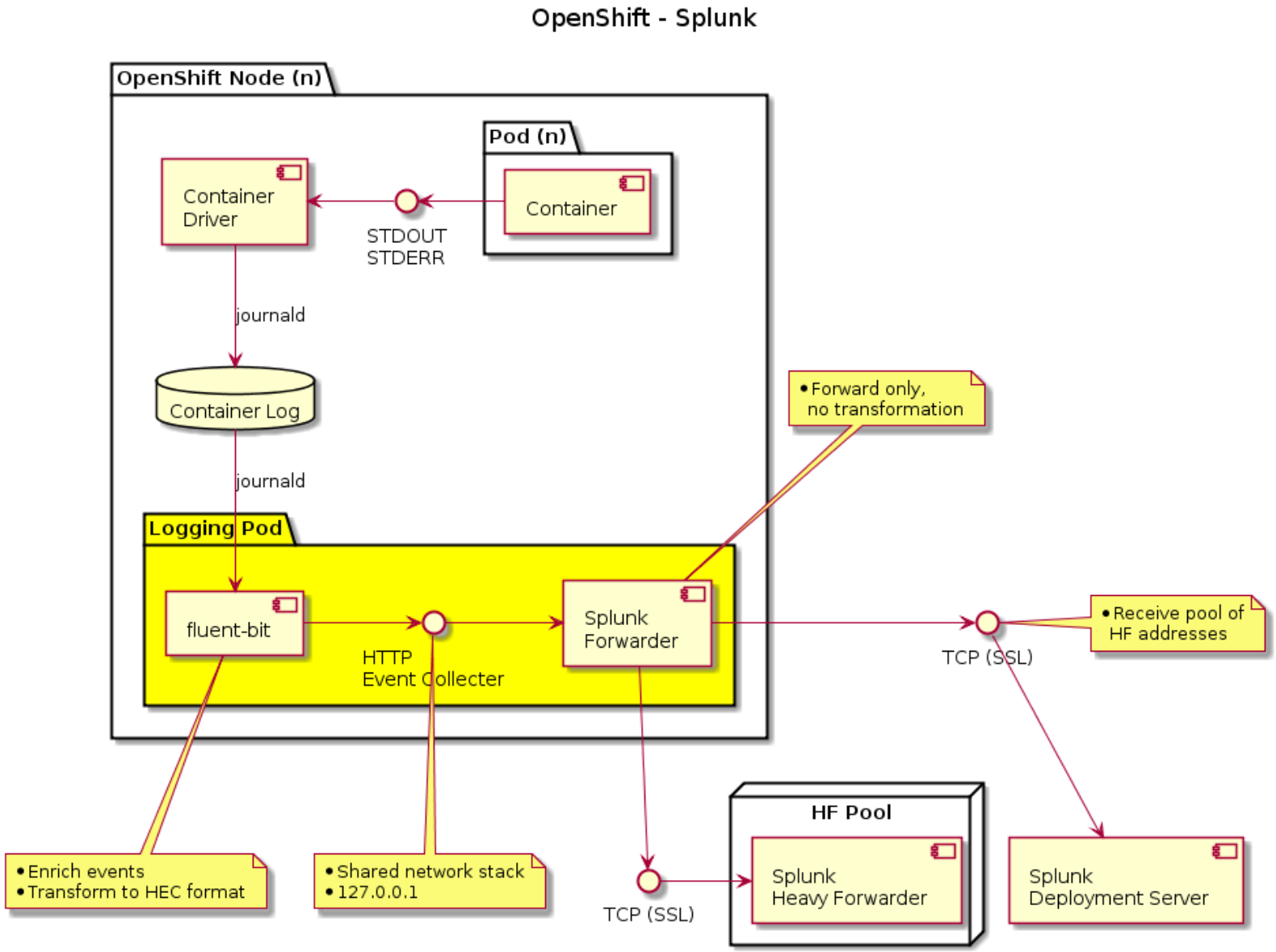

Figure 4: Components

Solution - sequence

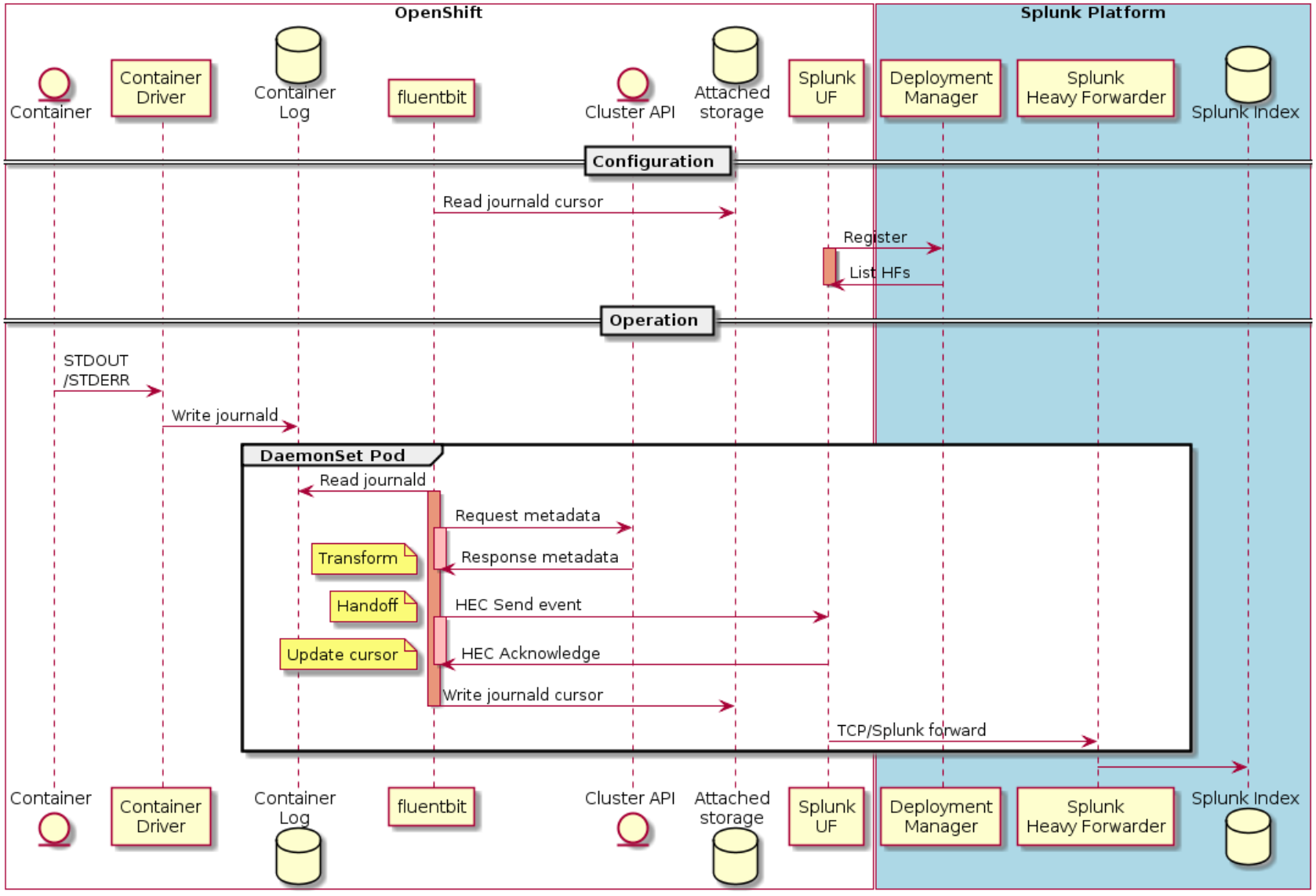

Figure 5: Sequence

fluent-bit pipeline

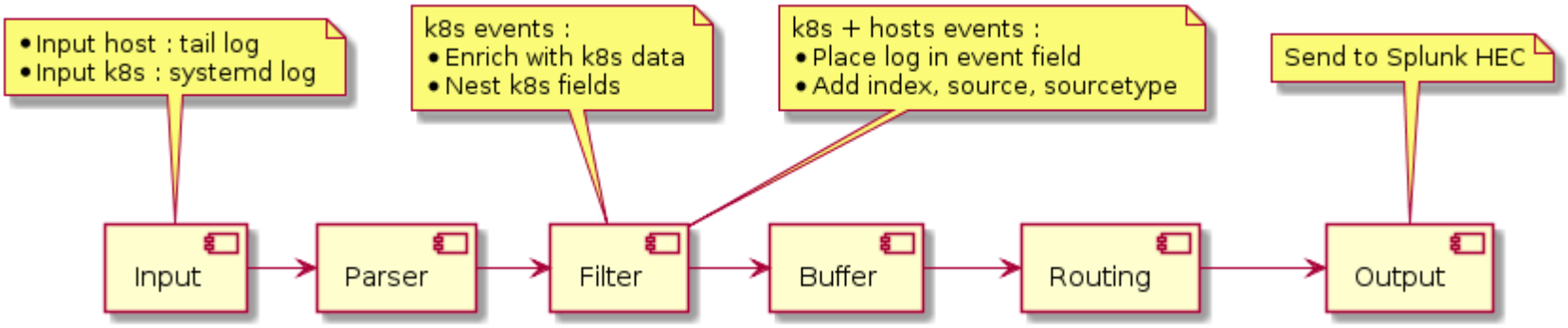

Figure 6: fluentbit pipepline

fluent-bit transformation

Input - log entry

```
[INFO] Application started ..
```

fluent-bit transformation

Output - HEC format, enriched

```
{
    "host"          : "service-01.prod.internal",
    "sourcetype"    : "structured_java_log_format_v3",
    "event"         : "[INFO] Application started ..",
    "fields"        : {
        "k8s:app_name"       : "service_app",
        "k8s:namespace_name": "production",
        "k8s:pod_id"         : "asdadsasd123",
        "k8s:container_id"   : "zxczxczxc456",
        "k8s:labels:app"     : "post_app"
    }
}
```

Feature - k8s enrichment - requirements

- Add all kubernetes metadata to each event
  - Clients cannot influence this
- Do not contaminate the source event
  - Meet audit requirements
- Do not overload storage
  - No duplication, use indexed fields

Feature - k8s enrichment - implementation

```
[FILTER]
    Name     nest
    Match    app.kubernetes.*

    # These become HEC indexed fields
    Operation    nest
    Nest_under   fields
    Wildcard     k8s*
```

Feature - self-service event routing - requirements

- Client control over event routing
  - Self-service requirement
- This is a platform service
  - No integration changes for applications

Feature - self-service event routing - k8s

Deployment YAML

```
template:
  metadata:
    labels:
      app: my-app
    annotations:
      index: com_myapp_logs
      sourcetype: myapp_java_v1
```

Feature - self-service event routing - integration

```
[FILTER]
    Name     modify
    Match    app.kubernetes.*
    # These become HEC top-level fields
    Rename   k8s:annotations:index index
    Rename   k8s:annotations:sourcetype sourcetype
```

Monitoring - metrics

`fluentbit` has can publish metrics in Prometheus format with metrics for individual inputs, filters and outputs.

```
$ curl localhost:2020/api/v1/metrics/prometheus
fluentbit_input_records_total{name="cpu.0"} 57
fluentbit_input_bytes_total{name="cpu.0"} 18069
fluentbit_output_proc_records_total{name="stdout.0"} 54
[...]
```

Monitoring - heartbeat

A heartbeat allows us to measure the health of the pipeline.

- Consistent rate
- Consistent volume

Useful for,

- Evaluating changes to the pipeline (QA)
- Monitoring rate/volume for issues with pipeline (alerting)

`fluentbit` inputs are backpressure sensitive, input slows down when there are upstream issues.

Monitoring - heartbeat

Configure a heartbeat input

```
[INPUT]
    Name    dummy
    Tag     sys.heartbeat
    Dummy   {"event":{"heartbeat":"heartbeat"}}
    Rate    1
```

Monitoring - heartbeat

Route the heartbeat to the correct index and have it passed up through the pipeline via the same channels as the log data

```
[FILTER]
    Name     modify
    Match    sys.heartbeat
    Set      sourcetype fluentbit-heartbeat
    Set      index fluentbit_hearbeat
```

Monitoring - reporting

```
rate(fluentbit_input_records_total{name=~"dummy.+"}[2m])
```

Figure 7: Grafana

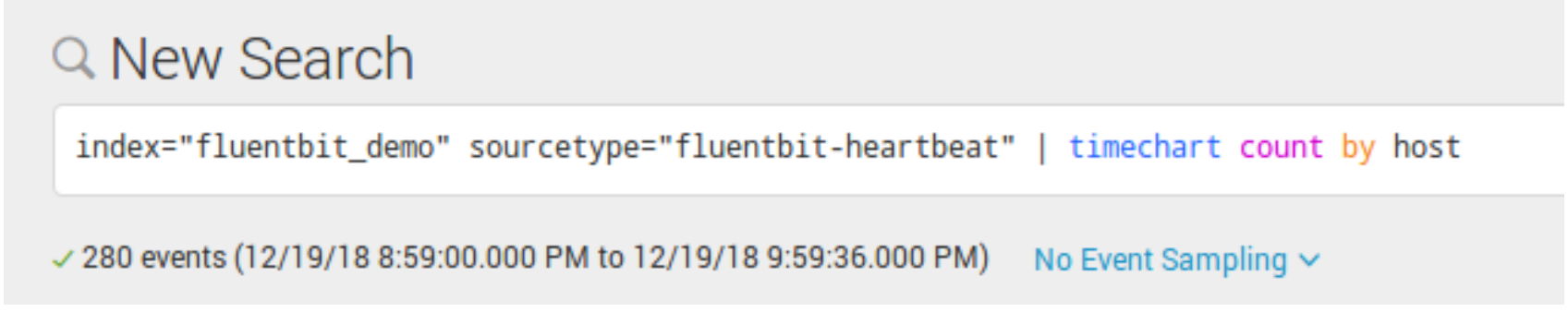

Figure 8: Splunk