Using Elasticsearch, Beats and Elastic APM to monitor your OpenShift Data

Michael Heldebrant

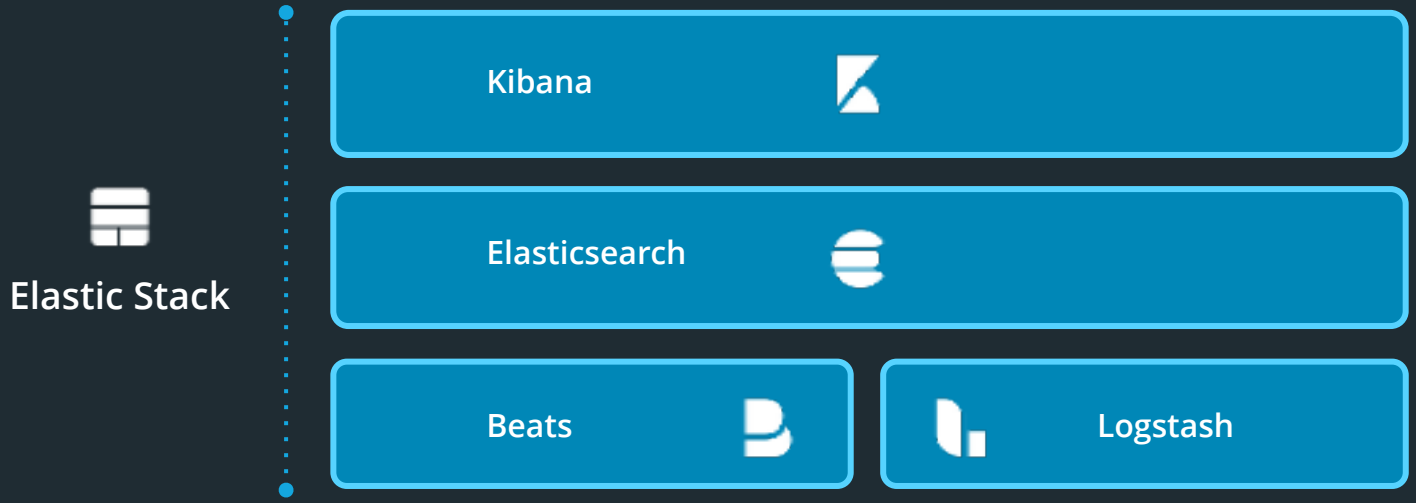
November 2018

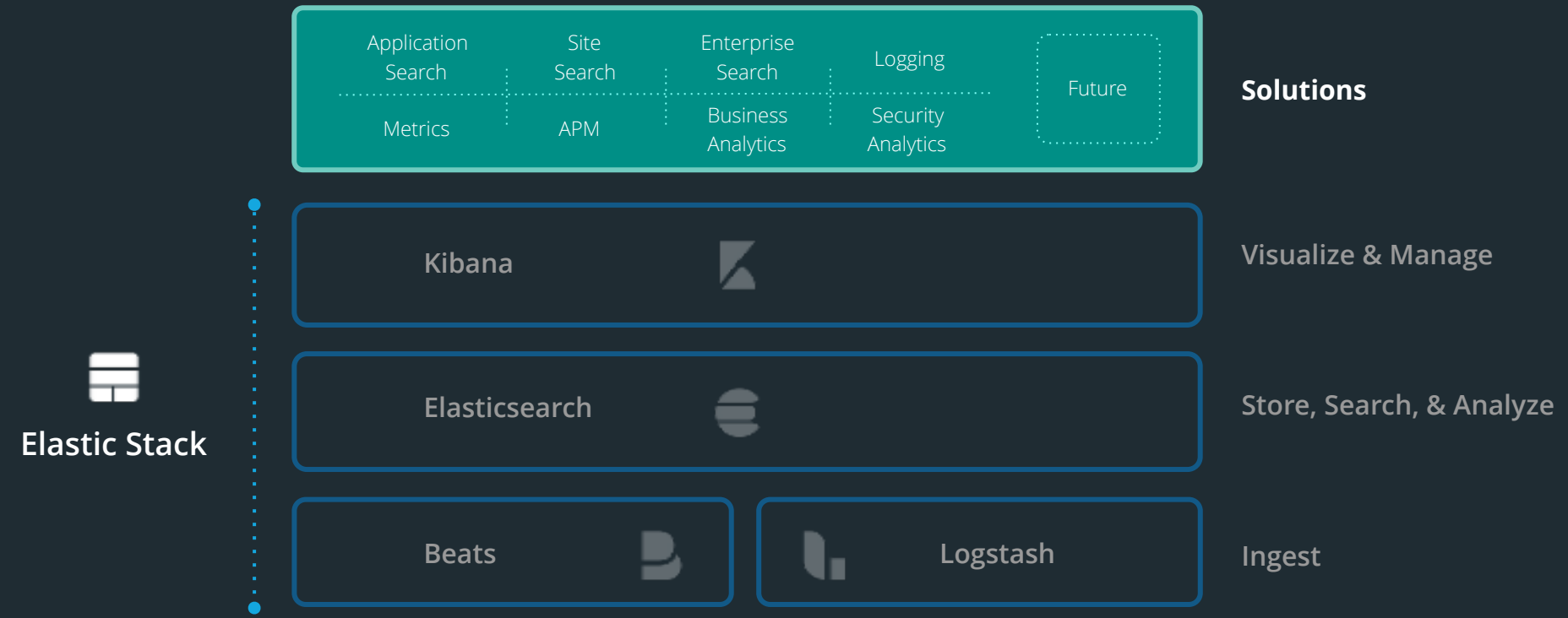
Michael Heldebrant
Solutions Architect
Elastic

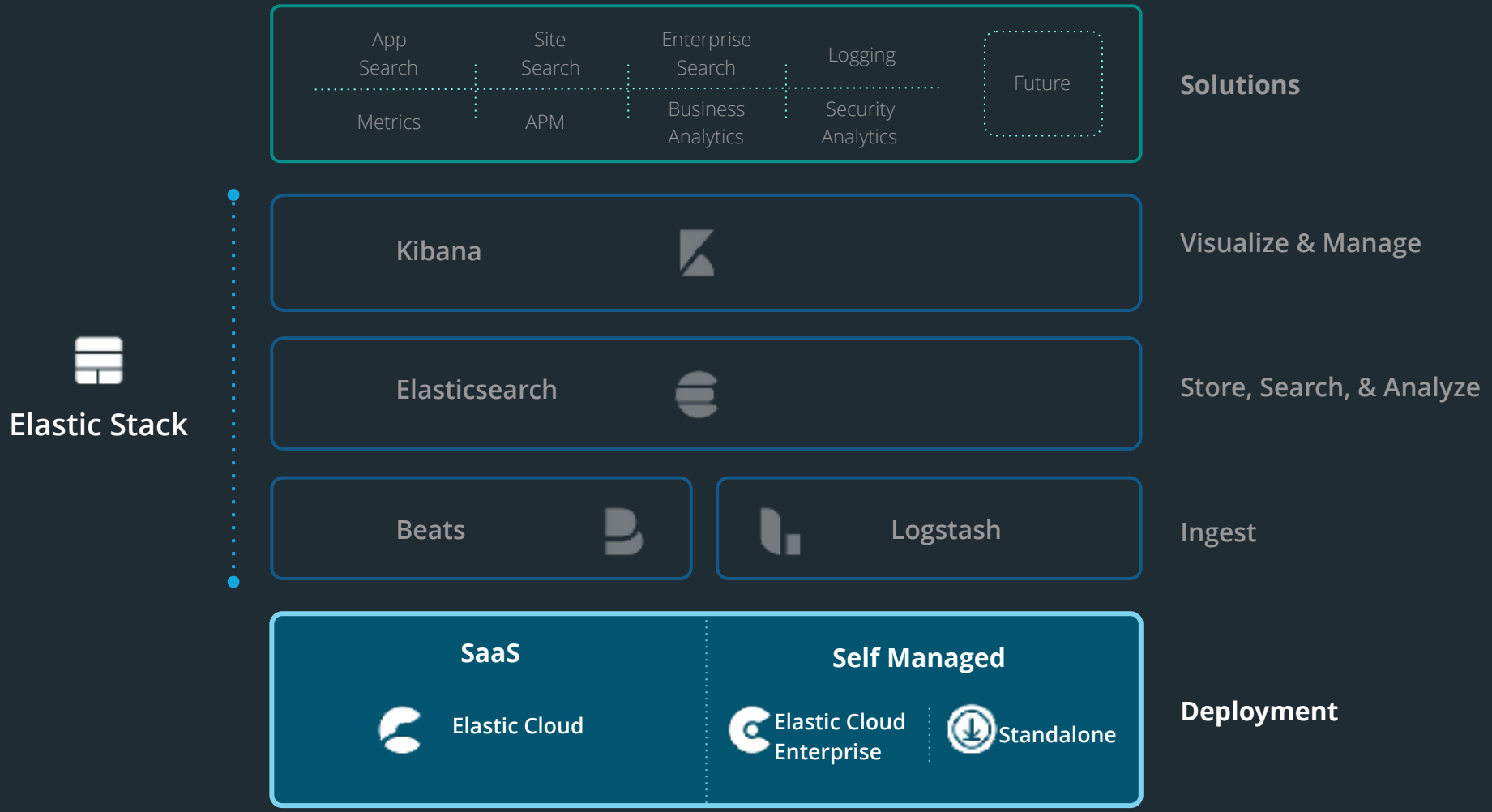
elastic

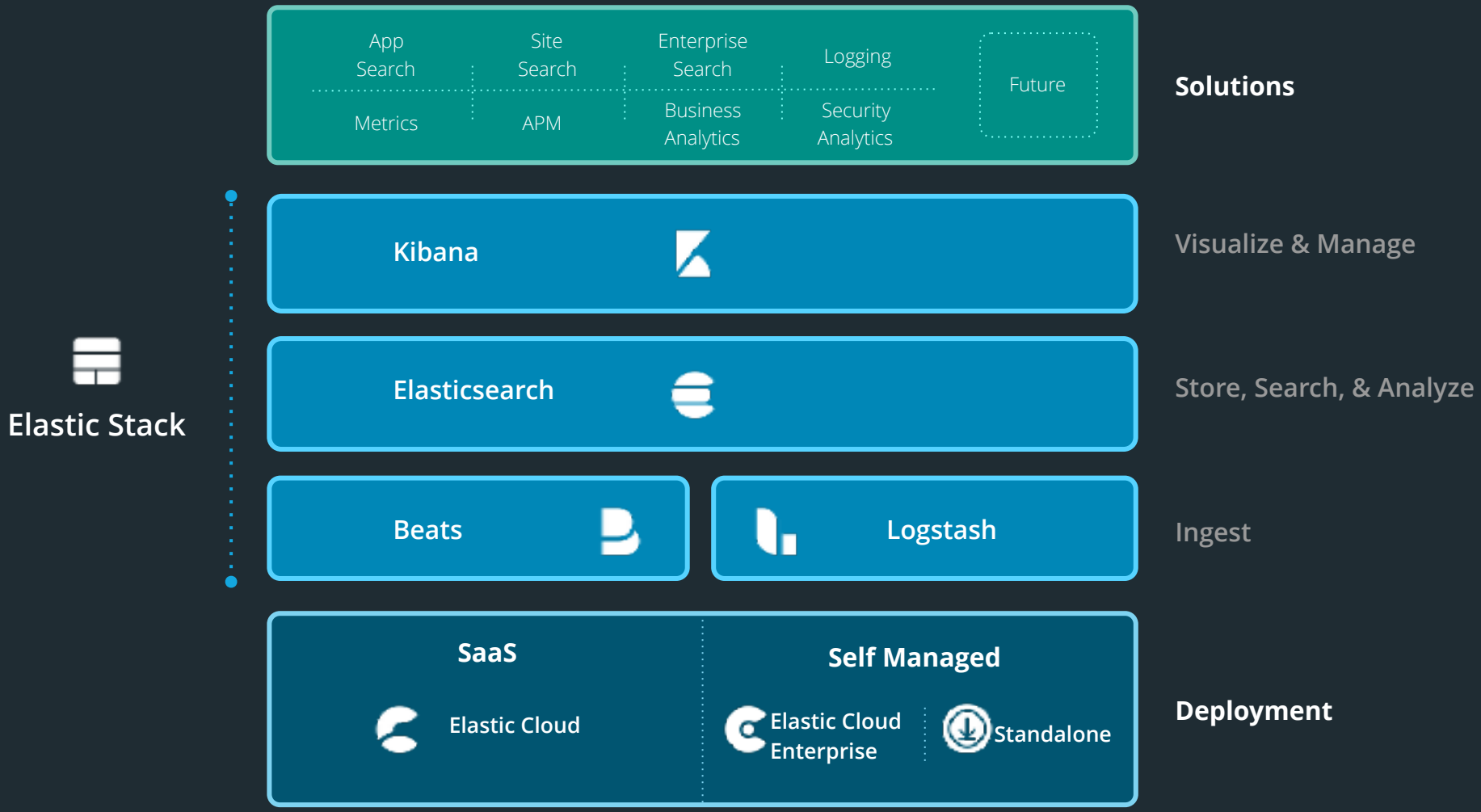
Housekeeping & Logistics

- **Slides and recording** will be available following the webinar
- Chat via IRC #elastic-webinar
  - #elastic-webinar @ Freenode
  - Click ”Join the Chat” link, create an IRC account
- Please select high resolution in the YouTube video player


Elastic Stack
Kibana
Visualize & Manage
Elasticsearch
Store, Search, & Analyze
Beats
Logstash
Ingest


Application Search
Site Search
Enterprise Search
Logging
Metrics
APM
Business Analytics
Security Analytics
Future
Solutions
Kibana
Visualize & Manage
Elasticsearch
Store, Search, & Analyze
Beats
Logstash
Ingest
Elastic Stack

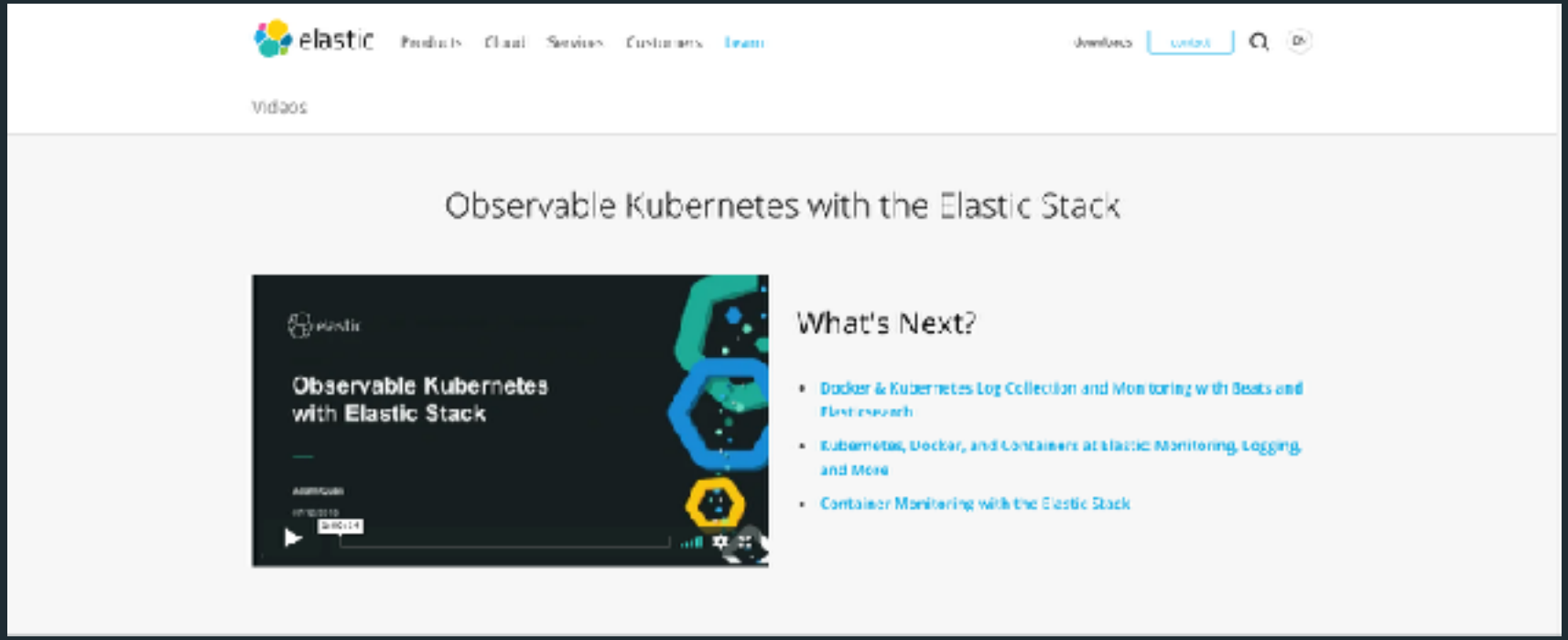

App Search
Site Search
Enterprise Search
Logging
Metrics
APM
Business Analytics
Security Analytics
Future
Solutions
Kibana
Visualize & Manage
Elasticsearch
Store, Search, & Analyze
Beats
Logstash
Ingest
Elastic Stack
SaaS
Elastic Cloud
Self Managed
Elastic Cloud Enterprise
Standalone
Deployment


App Search
Site Search
Enterprise Search
Logging
Metrics
APM
Business Analytics
Security Analytics
Future
Solutions
Kibana
Visualize & Manage
Elasticsearch
Store, Search, & Analyze
Beats
Logstash
Ingest
Elastic Stack
SaaS
Elastic Cloud
Self Managed
Elastic Cloud Enterprise
Standalone
Deployment

Agenda

- Three pillars of observability
- OpenShift compared to Kubernetes
- Elastic Beats and OpenShift
- APM tracing
- Demo

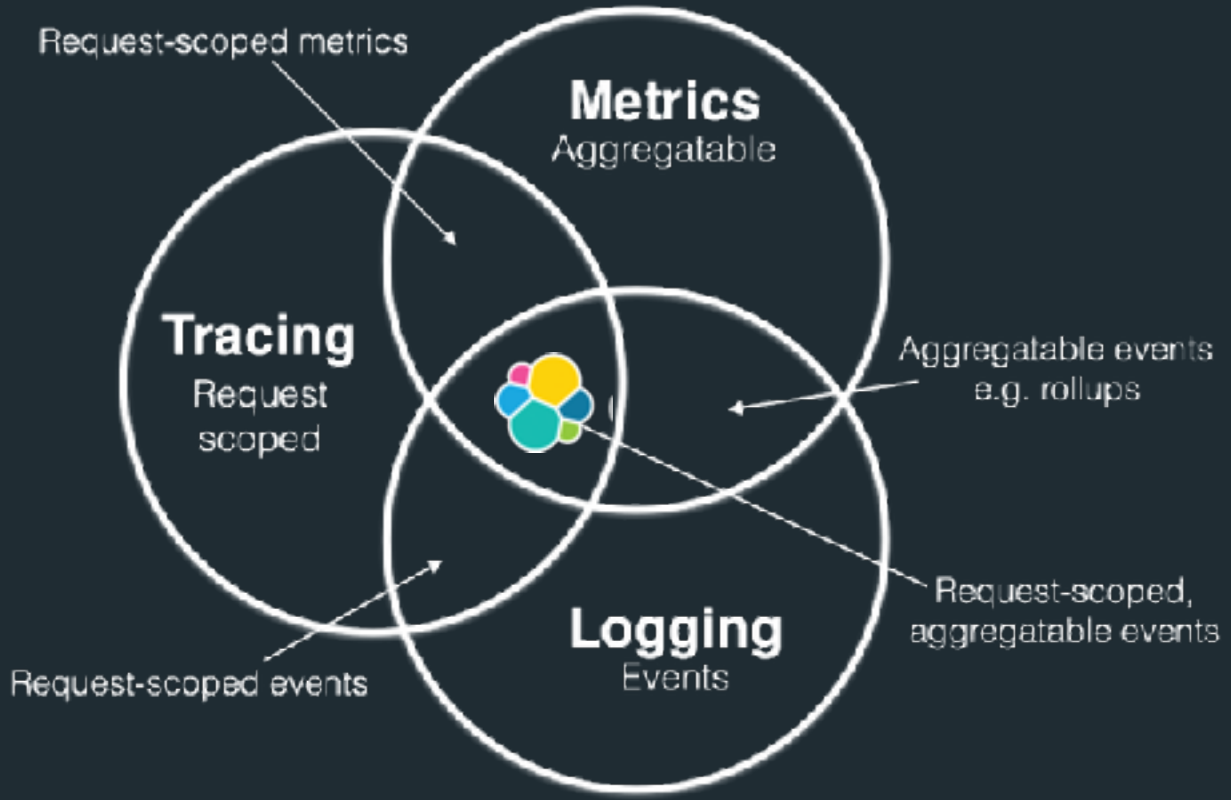
Kubernetes webinar



It Comes Down to The Three Pillars of Observability

Twitter:
https://blog.twitter.com/engineering/en_us/a/2013/observability-at-twitter.html
Peter Bourgon
https://peter.bourgon.org/blog/2017/02/21/metrics-tracing-and-logging.html

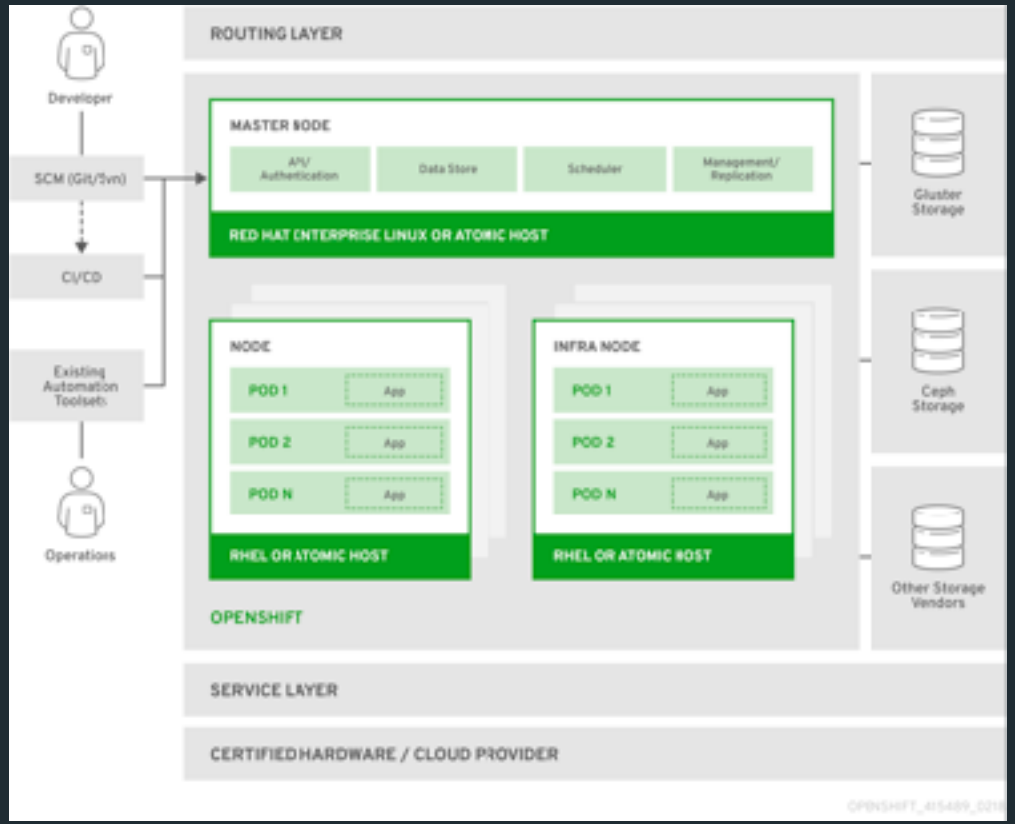
Elastic at the Center Stage



OpenShift adds on to Kubernetes

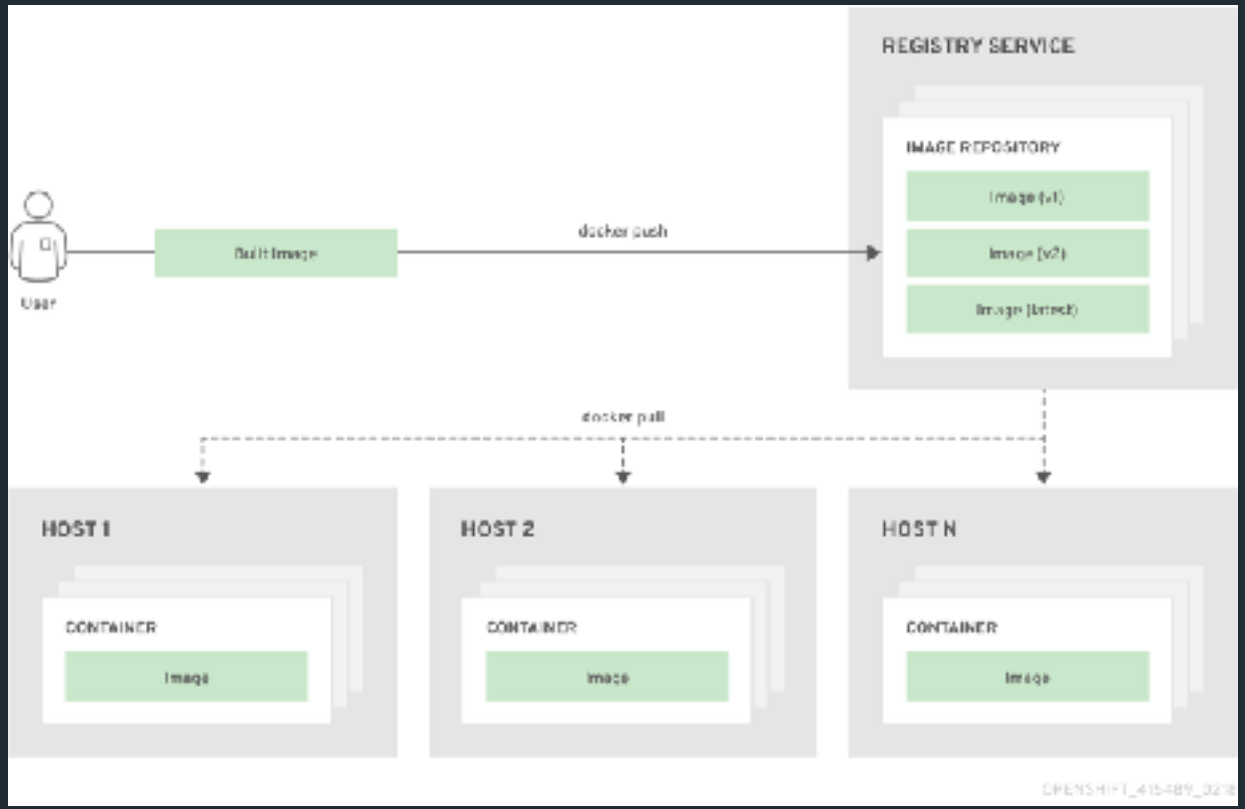
- Pod to Pod network stack (openvswitch and vxlan)
- Router (based on HAProxy)
- Docker registry
- Source to Image builds (from source code to deployed applications)
- Security Context Controls (privileged containers, selinux)
- OpenShift Web Console

OpenShift High Level Architecture



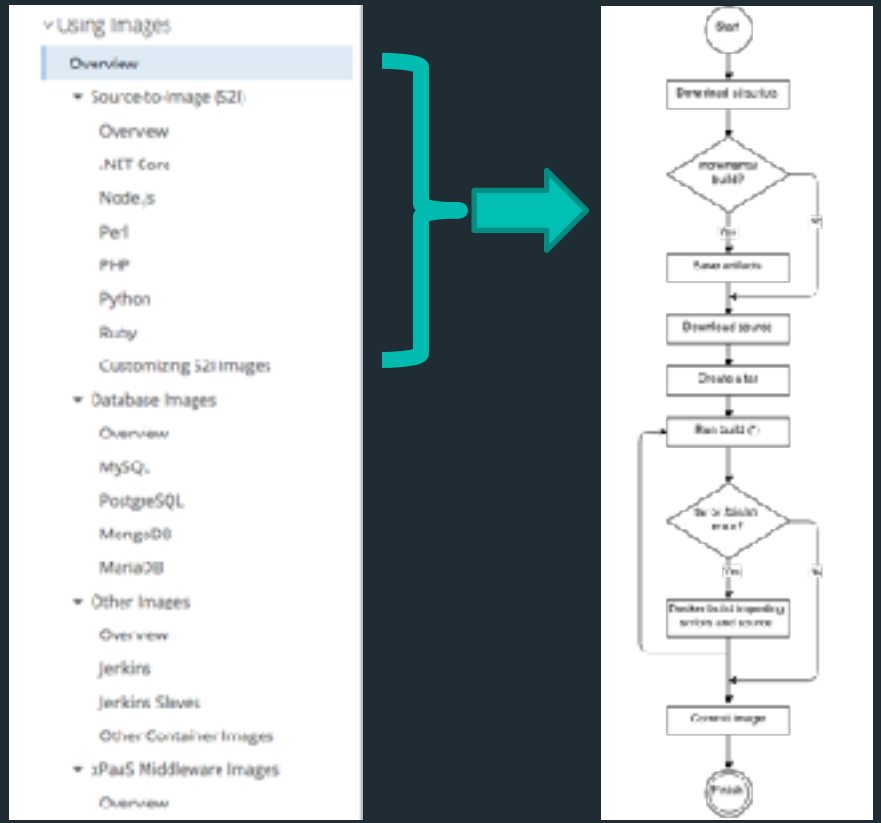
https://access.redhat.com/documentation/en-us/openshift_container_platform/3.11/

OpenShift – bring your own containers



https://access.redhat.com/documentation/en-us/openshift_container_platform/3.11/

OpenShift – build your own containers via S2I

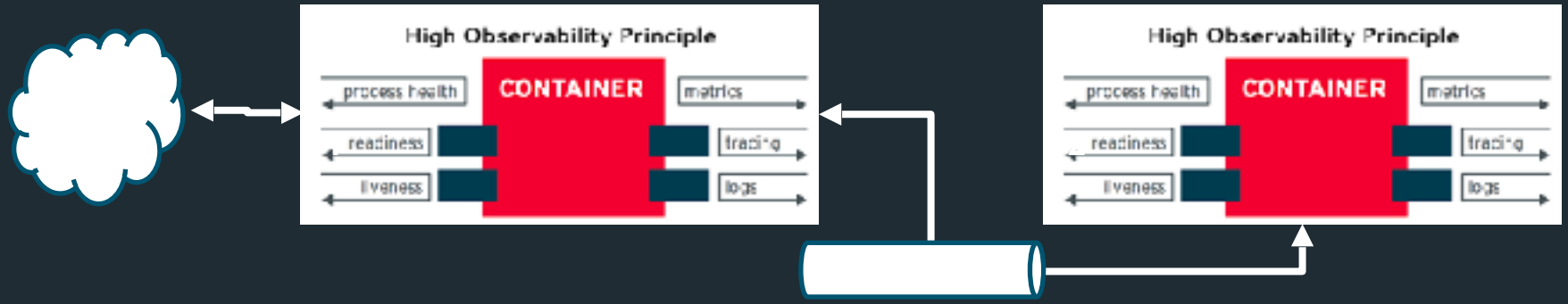


https://docs.openshift.com/container-platform/3.11/using_images/index.html

Principles of container-based application design

https://www.redhat.com/en/resources/cloud-native-container-design-whitepaper

What is possible today



- Logs: Filebeat collecting logs and sending to ingest pipelines for log analytics
- Health/Metrics: Metricbeat collecting metrics from the infrastructure, applications, and platform
- Tracing: APM agents instrumenting application code
- Readiness: Heartbeat to monitor uptime and latency from external hosts
- Network: Packetbeat analyzing pod to pod traffic

Technical Details of the demo

- **Filebeat** as a daemonset on each node for pods (system logs also possible)
- **Metricbeat** as a daemonset on each node for system and pod metricsets
- **Metricbeat** as a deployment collecting from kube-state metrics deployment
- **Elastic APM** agents sending to an **apm-server** package installed on the OS
- **Packetbeat** package installed on the OS for pod network traffic monitoring
- **Heartbeat** package installed on external OS host for service monitoring
- **All data sent to Elastic Cloud Elasticsearch Service** cluster and **Kibana**

High level Steps for Beats running in Openshift

```
Need a Cluster? – Elastic Cloud Elasticsearch Service 14 day trial
```

Get the Filebeat manifest, modify as per documentation, oc create, and grant scc to service account

- **https://www.elastic.co/guide/en/beats/filebeat/6.5/running-on-kubernetes.html)**

Get the Kube-state-metrics manifest, oc create

Get the Metricbeat manifest, modify as per documentation, oc create, and grant scc to service account

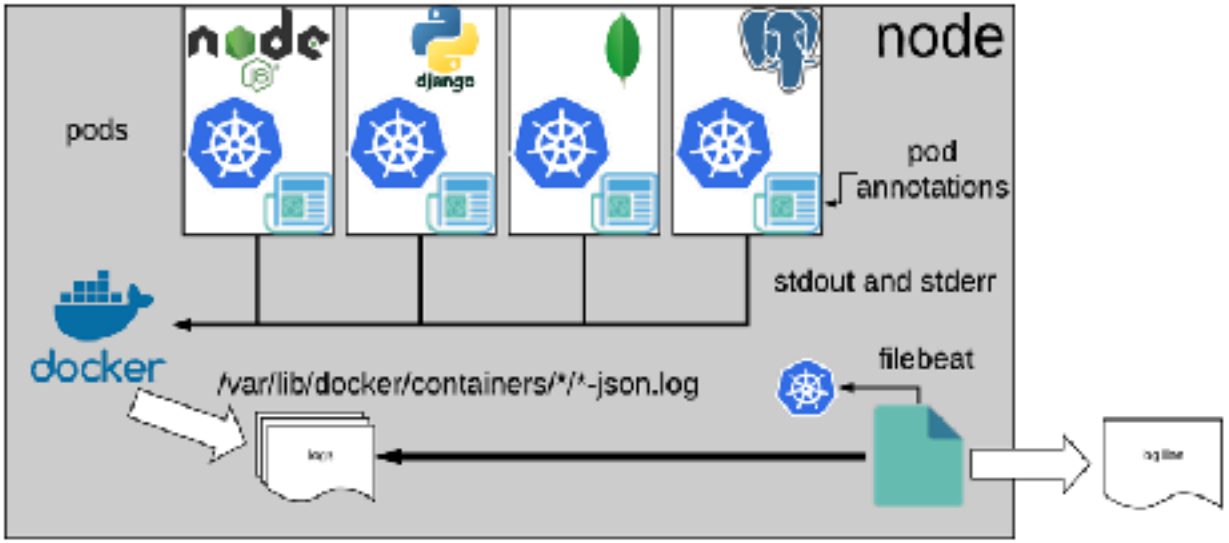
- **https://www.elastic.co/guide/en/beats/metricbeat/6.5/running-on-kubernetes.html**

What changes are needed compared to Kubernetes?

- **Filebeat** needs to run as a privileged container to mount logs written on node (hostPath) and read them (User: RunAsAny and selinux: RunAsAny)
- Metricbeat needs to run as a privileged container to attach to the host network (Allow Host Network) and mount system volumes (hostPath) and read them (User: RunAsAny and selinux: RunAsAny). Metricbeat needs to use a bearer token and RBAC for the secured kubelet port for metrics collection.
- The Service accounts for Filebeat and Metricbeat need to be added to the privileged scc. This is the same access required for other log shippers.

```
oc adm policy add-scc-to-user privileged system:serviceaccount:kube-system:filebeat
oc adm policy add-scc-to-user privileged system:serviceaccount:kube-system:metricbeat
```

- # Filebeat: Lightweight Shipper for Logs



- Filebeat daemon set per node
- Mounts filesystem from node
- Reads log files and adds metadata
- Sends json to ingest pipeline for parsing based on pod annotations

Get multiline (ie stacktrace) support per pod and per container in a pod

Hints based autodiscover

Filebeat supports autodiscover based on hints from the provider. The hints system looks for hints in Kubernetes Pod annotations or Docker labels that have the prefix `co.elastic.logs`. As soon as the container starts, Filebeat will check if it contains any hints and launch the proper config for it. Hints tell Filebeat how to get logs for the given container. By default logs will be retrieved from the container using the docker input. You can use hints to modify this behavior. This is the full list of supported hints:

`co.elastic.logs/disable`

Filebeat gets logs from all containers by default, you can set this hint to `true` to ignore the output of the container. Filebeat won't read or send logs from it.

`co.elastic.logs/multiline.*`

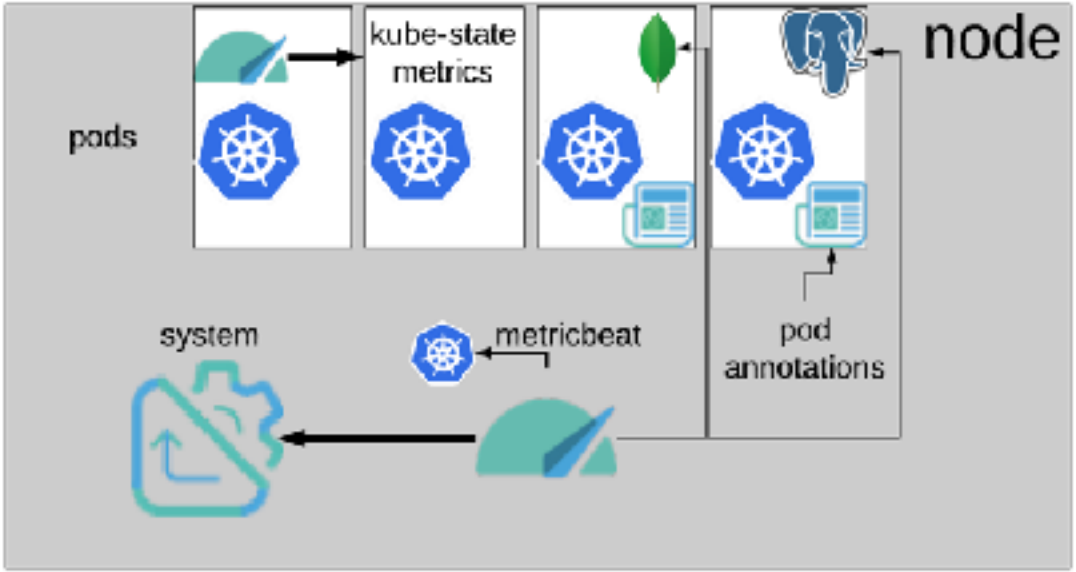
Multiline settings. See *Manage multiline messages* for a full list of all supported options.

Multiple containers

When a pod has multiple containers, the settings are shared unless you put the container name in the hint. For example, these hints configure multiline settings for all containers in the pod, but set a specific `exclude_lines` hint for the container called `sidecar`.

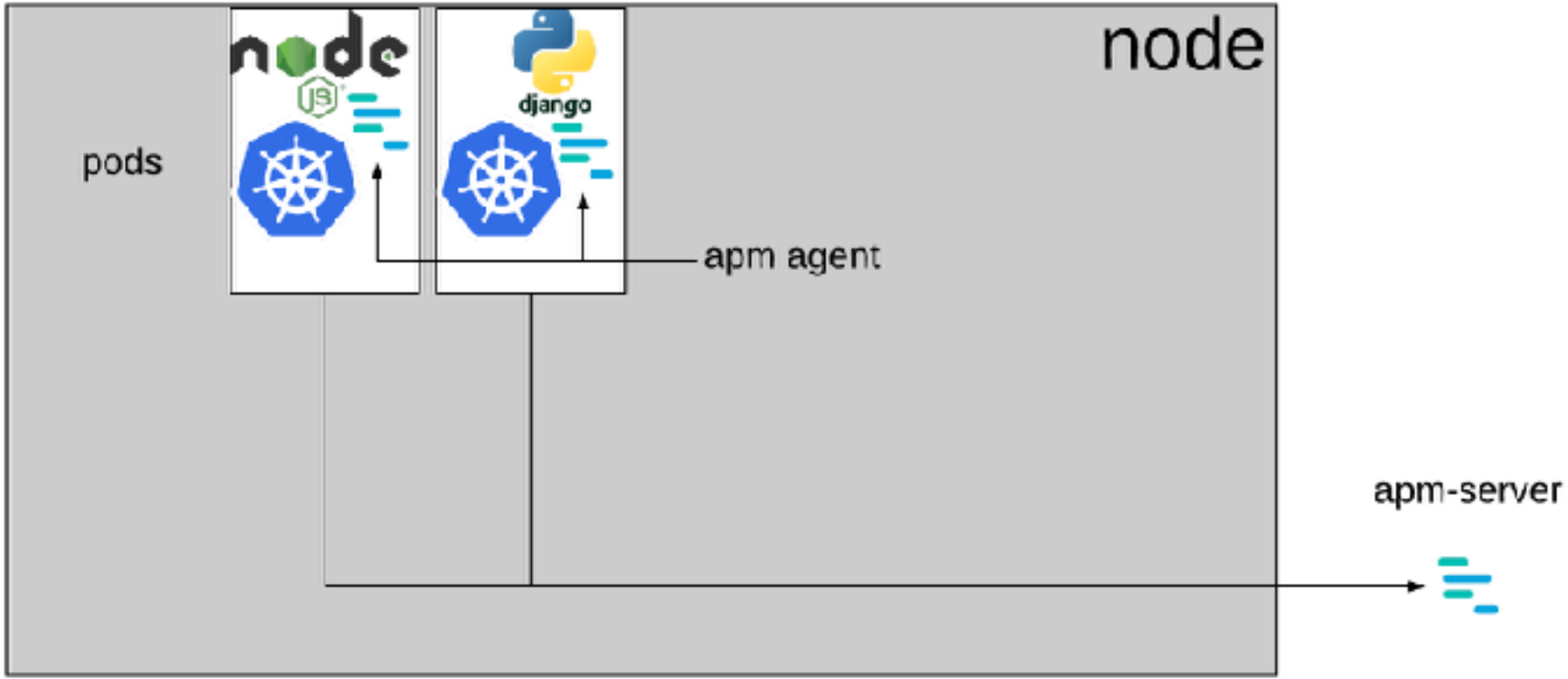
```
annotations:
  co.elastic.logs/multiline.pattern: '^\['
  co.elastic.logs/multiline.negate: true
  co.elastic.logs/multiline.match: after
  co.elastic.logs.sidecar/exclude_lines: '^DBG'
```

Metricbeat: Lightweight Shipper for Metrics

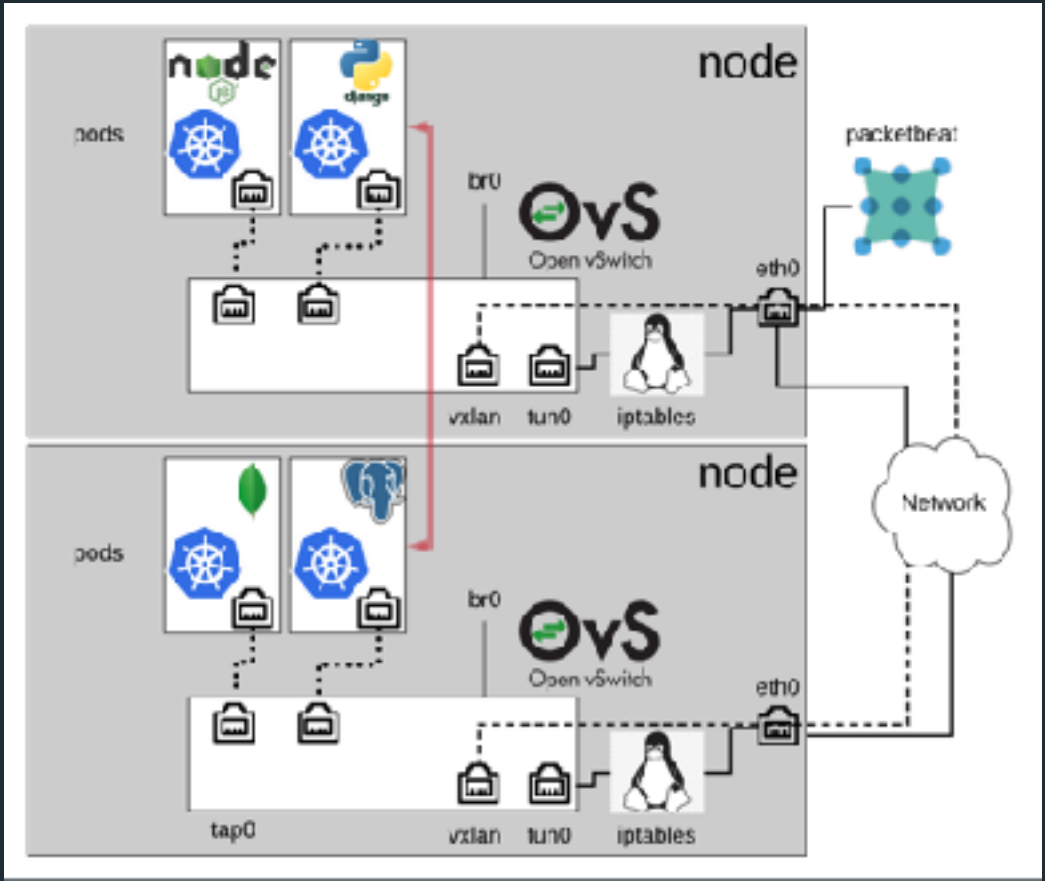


- Metricbeat daemon set per node
- Mounts filesystem from node
- Collects host and process metrics, also can collect Prometheus metrics
- Metricbeat Deployment to collect kube-state-metrics data

Elastic APM: Open Source Application Monitoring



Packetbeat: Lightweight Shipper for Network Data
On eth0 sees pod to pod traffic as port 4789 UDP (vxlan)



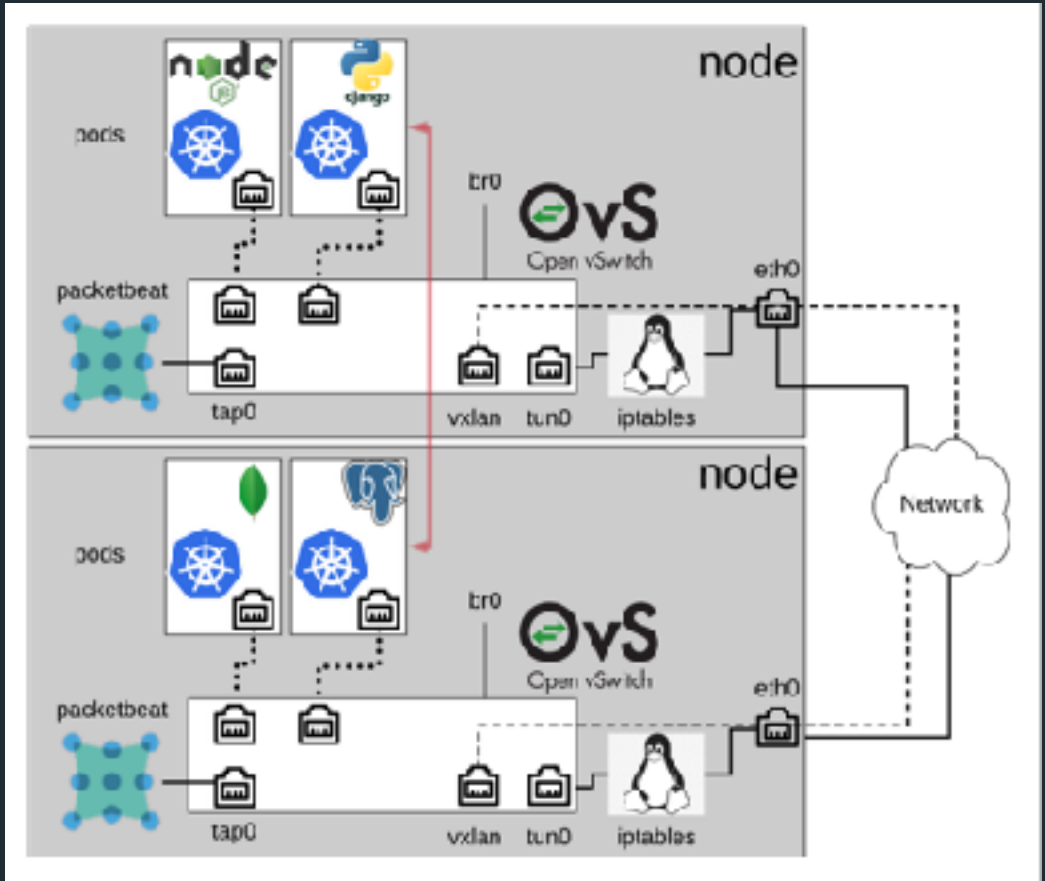
Create a span port via openvswitch

```
#Create port for sdn network tap
ovs-vsctl add-port br0 tap0 -- set interface tap0 type=internal
#Mirror all traffic to the tap port
ovs-vsctl -- set Bridge br0 mirrors=@m -- --id=@tap0 get Port tap0 -- --
id=@m create Mirror name=mymirror select-all=true output-port=@tap0
```

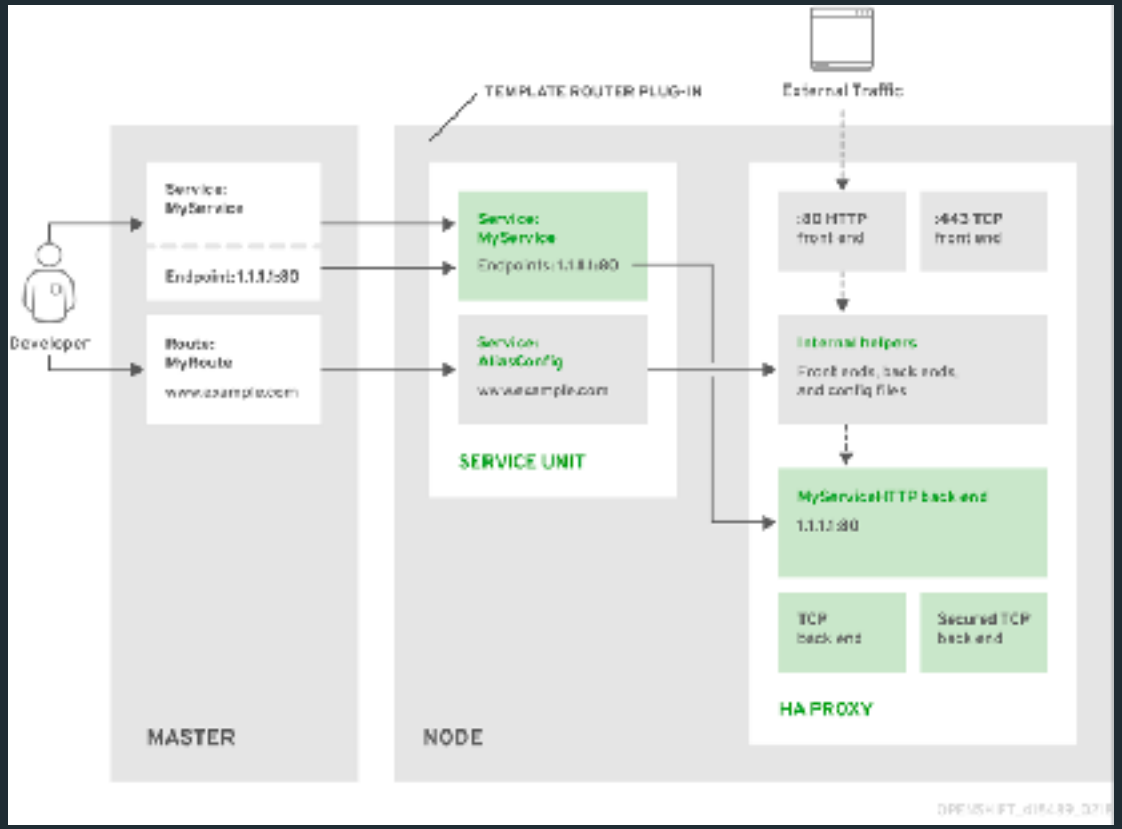
packetbeat.yml

```
# Select the network interface to sniff the data. On Linux, you can use the
# "any" keyword to sniff on all connected interfaces.
packetbeat.interfaces.device: tap0
```

Packetbeat now can observe the pod to pod traffic



OpenShift router high level architecture



https://access.redhat.com/documentation/en-us/openshift_container_platform/3.11/

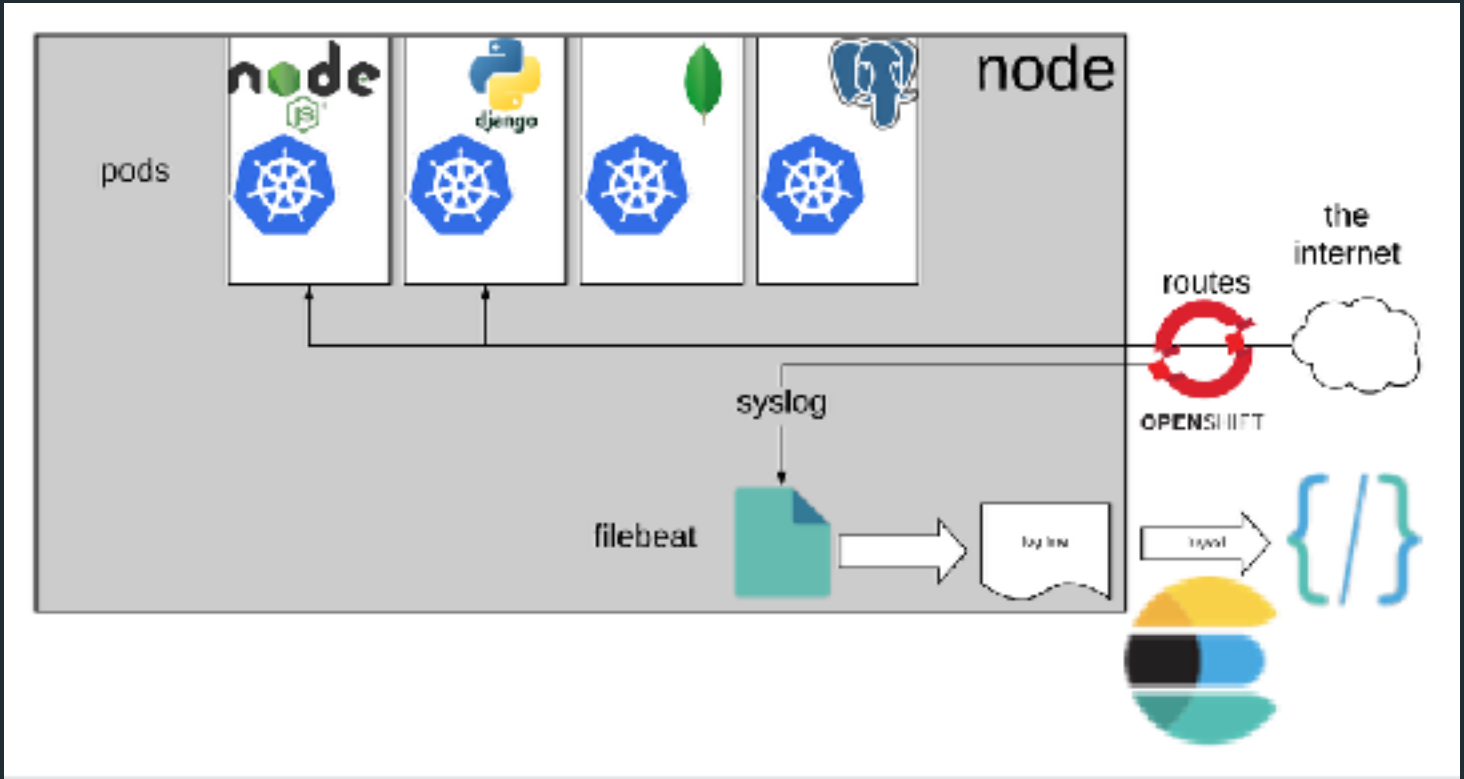
Send HAProxy logs out via syslog and collect via filebeat

```
$ oc set env dc/router ROUTER_SYSLOG_ADDRESS=127.0.0.1 ROUTER_LOG_LEVEL=debug
```

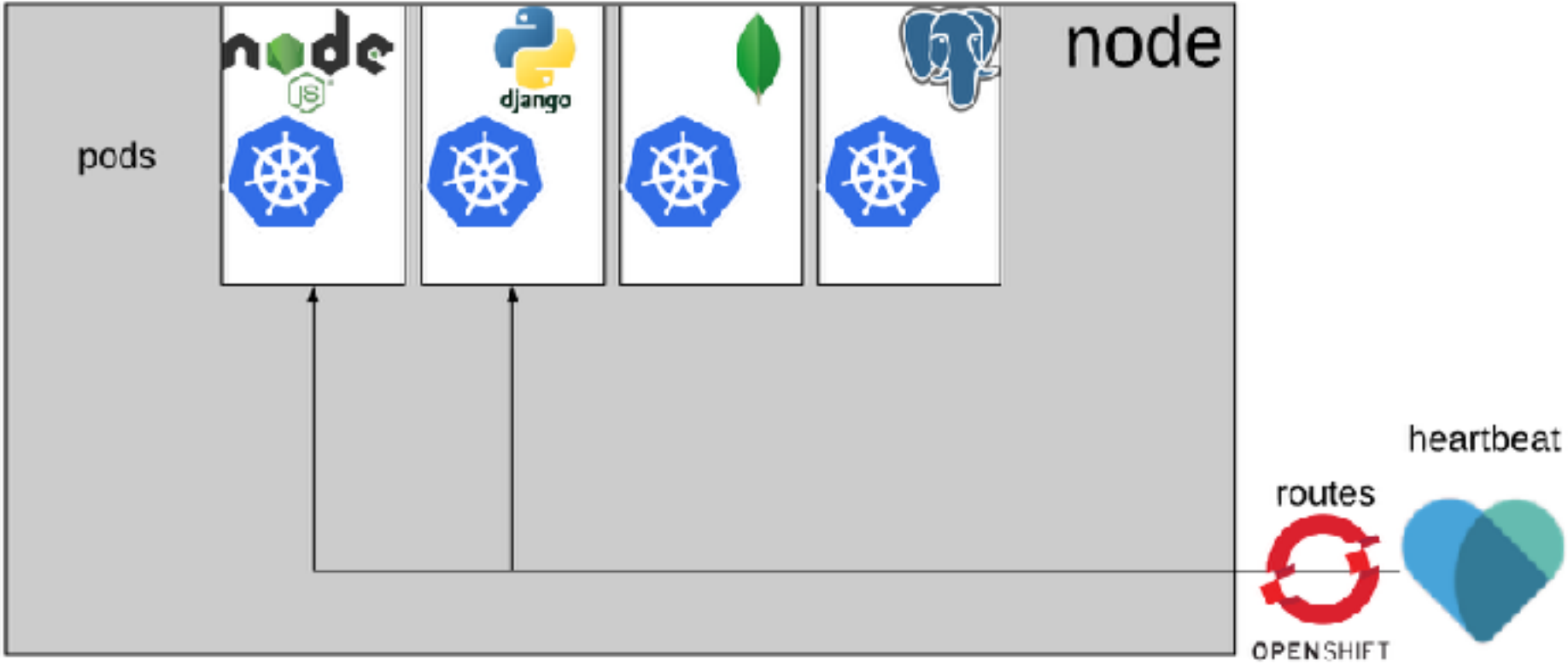
```
filebeat.yml

filebeat.inputs:
- type: syslog
  protocol.udp:
    host: "localhost:514"
```

HAProxy log analytics in an afternoon



Heartbeat: Lightweight Shipper for Uptime Monitoring



Demonstration

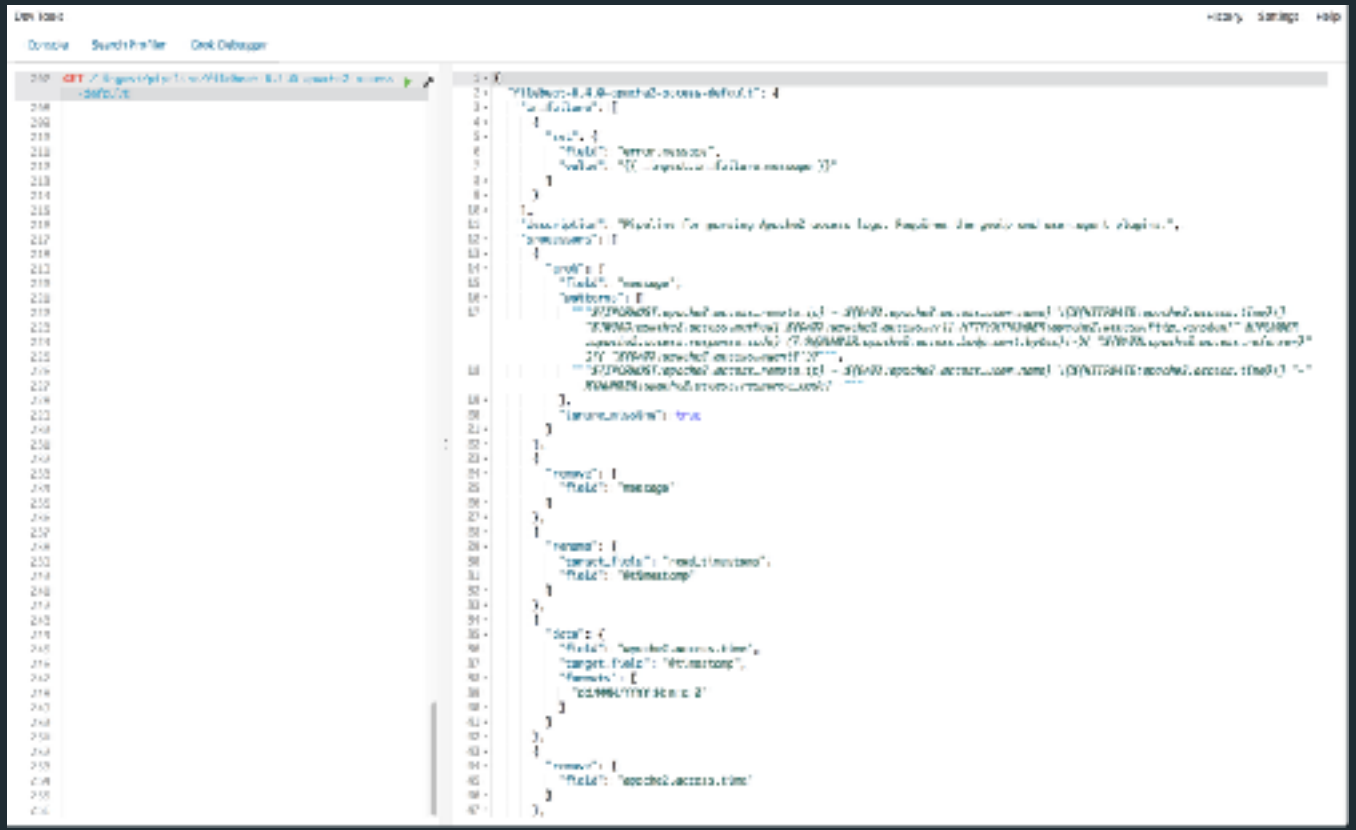
Questions?

Thank You

- **Web** : www.elastic.co
- Demos: demo.elastic.co
- **Products** : https://www.elastic.co/products
- **Forums** : https://discuss.elastic.co/
- **Community** : https://www.elastic.co/community/meetups
- **Twitter** : @elastic

Bonus slides

Start with an existing analogous ingest pipeline: apache2

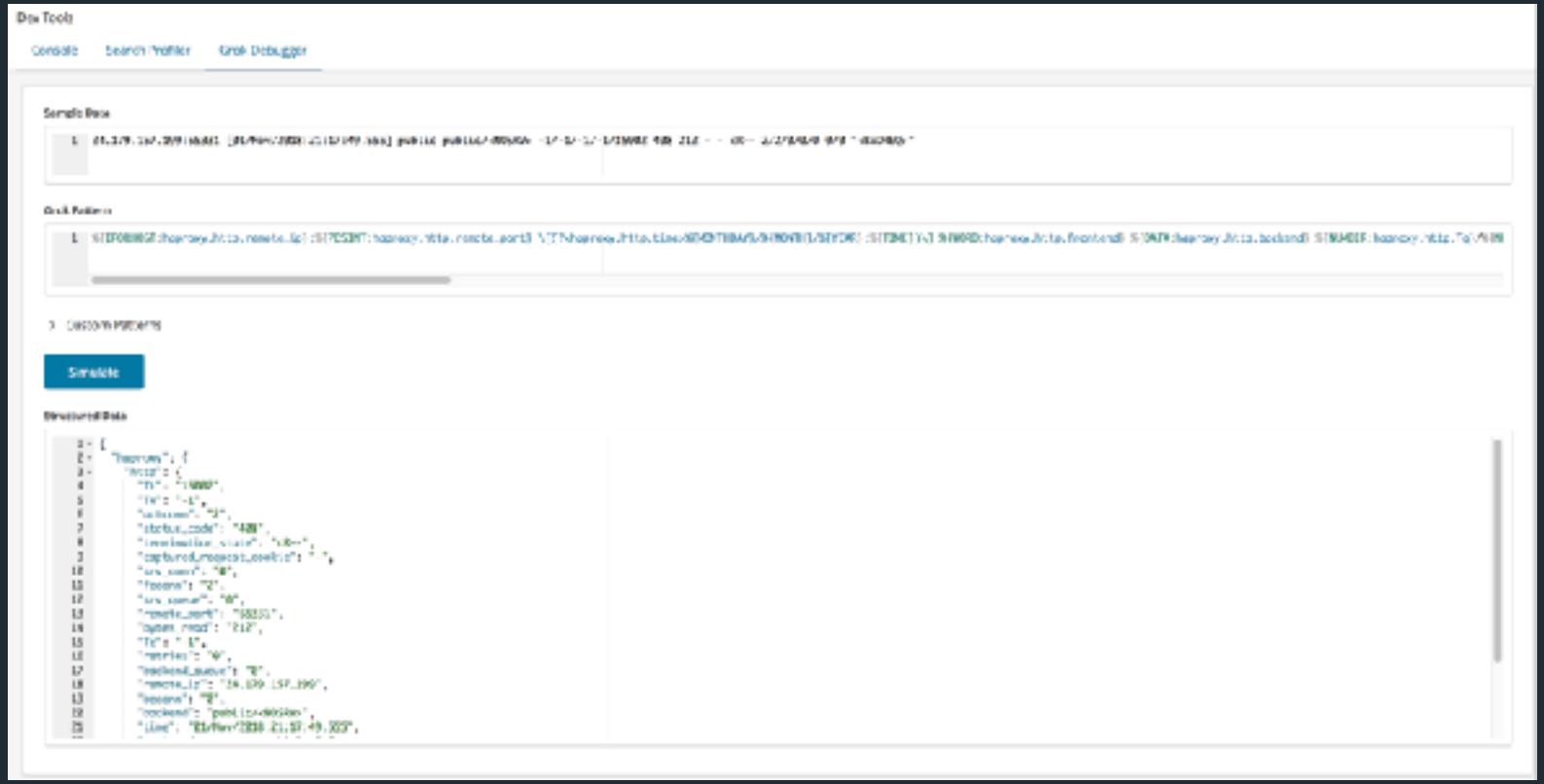


Modify the grok pattern for the HAProxy log format

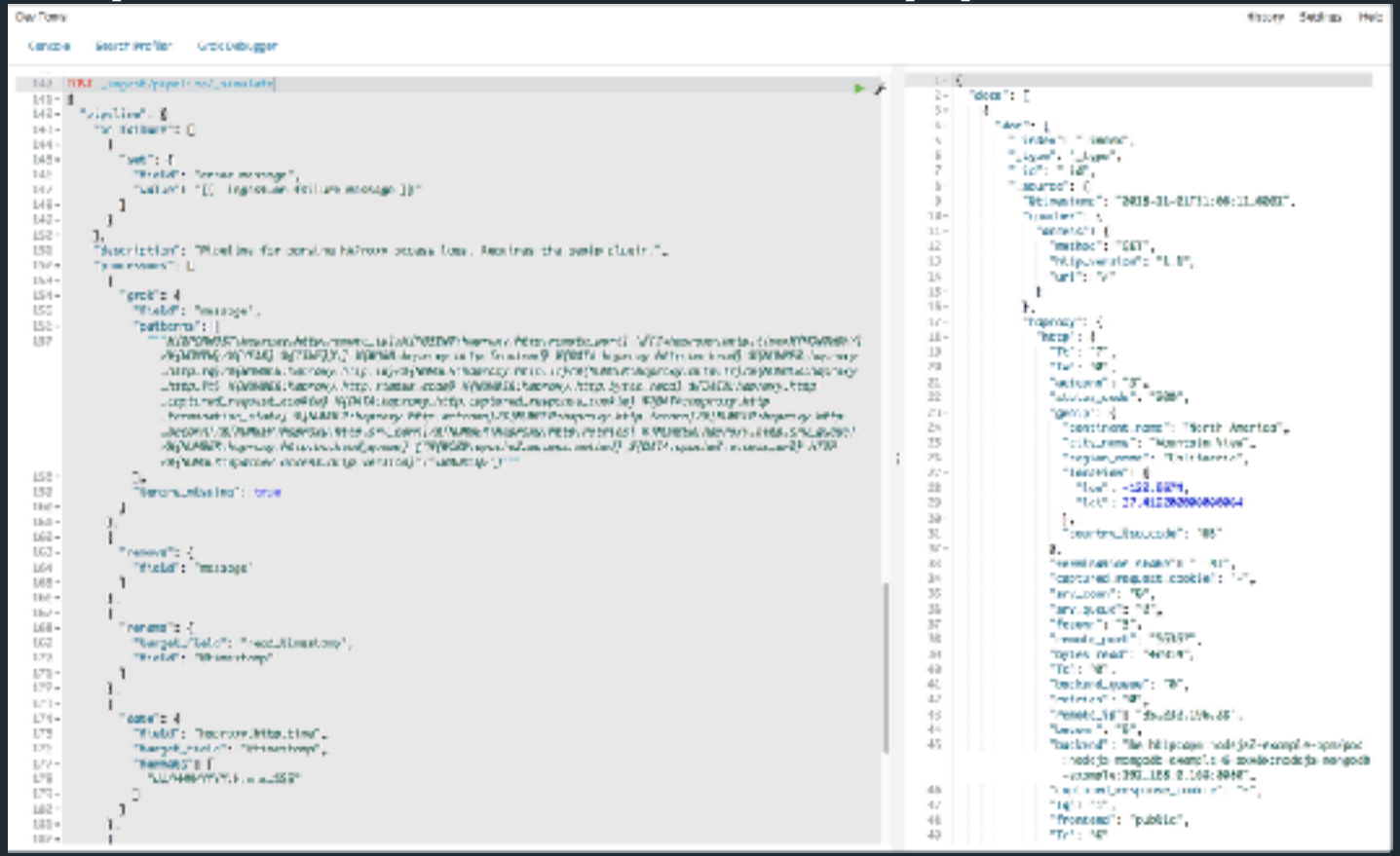
>>> Feb 6 12:14:14 localhost \ haproxy[14389]: 10.0.1.2:33317 [06/Feb/2009:12:14:14.655] http-in \ static/srv1 10/0/30/69/109 200 2750 - - ---- 1/1/1/1/0 0/0 {1wt.eu} \ {} "GET /index.html HTTP/1.1"

Field	Format	Extract from the example above
1	process_name '[' pid ']:'	haproxy[14389]:
2	client_ip ':' client_port	10.0.1.2:33317
3	'[' accept_date ']'	[06/Feb/2009:12:14:14.655]
4	frontend_name	http-in
5	backend_name '/' server_name	static/srv1
6	Tq '/' Tw '/' Tc '/' Tr '/' Tt*	10/0/30/69/109
7	status_code	200
8	bytes_read*	2750
9	captured_request_cookie	-
10	captured_response_cookie	-
11	termination_state	----
12	actconn '/' feconn '/' beconn '/' srv_conn '/' retries*	1/1/1/1/0
13	srv_queue '/' backend_queue	0/0
14	'{' captured_request_headers* '}'	{haproxy.1wt.eu}
15	'{' captured_response_headers* '}'	{}
16	'"' http_request '"'	"GET /index.html HTTP/1.1"

Grok Debugger – iterate till you parse it

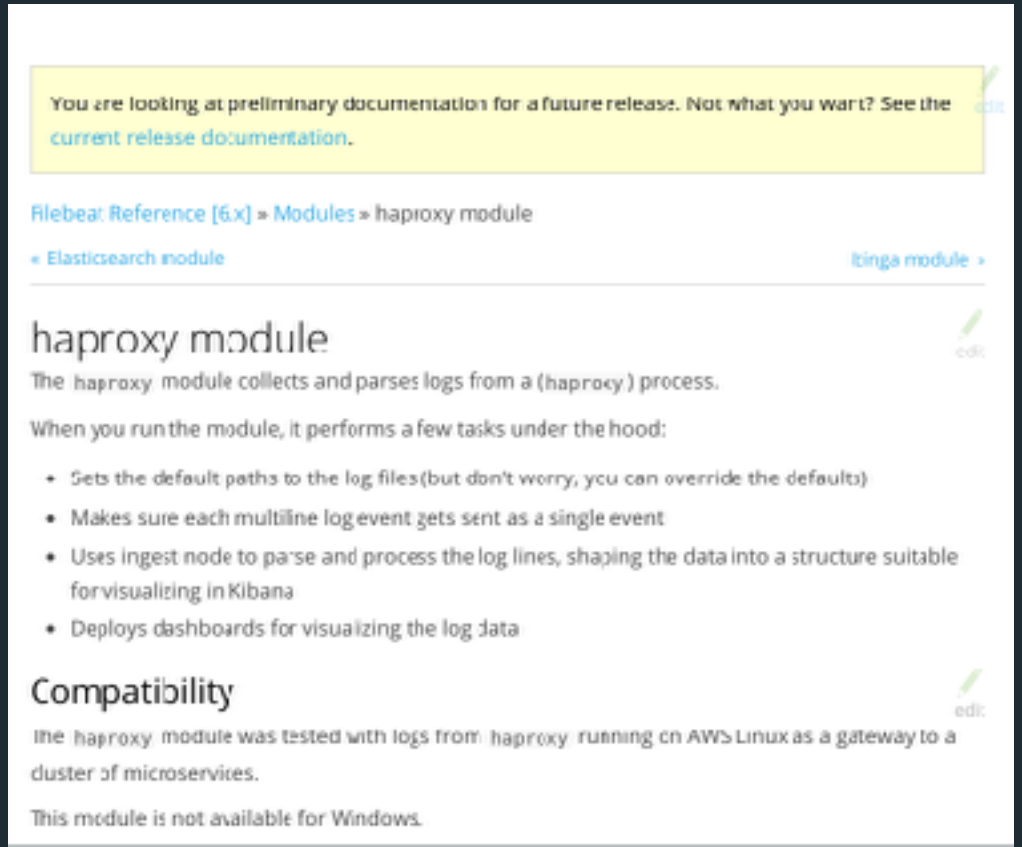


Ingest Pipeline Simulator – test full pipeline



Add a few mappings to the template

Then find out it's coming as a module in a future version



You are looking at preliminary documentation for a future release. Not what you want? See the current release documentation.

Filebeat Reference [6.x] » Modules » haproxy module

« Elasticsearch module Icinga module »

haproxy module

The `haproxy` module collects and parses logs from a (`haproxy`) process.

When you run the module, it performs a few tasks under the hood:

- Sets the default paths to the log files (but don't worry, you can override the defaults)
- Makes sure each multiline log event gets sent as a single event
- Uses ingest node to parse and process the log lines, shaping the data into a structure suitable for visualizing in Kibana
- Deploys dashboards for visualizing the log data

Compatibility

The `haproxy` module was tested with logs from `haproxy` running on AWS Linux as a gateway to a cluster of microservices.

This module is not available for Windows.