



Kubernetes on AWS: Observability



Special OFFERS



**Free Observability
Assessment**
for all eligible attendees



Kubernetes on AWS: Observability

PRESENTERS



Curtis Rissi

Senior Solutions Architect



Marius Ducea

VP of DevOps Practice



Gagan Goswami

DevOps Engineer





Amazon Web Services (AWS) is the **world's most comprehensive and broadly adopted cloud platform**, offering over 175 fully featured services from data centers globally. Millions of customers—including the fastest-growing startups, largest enterprises, and leading government agencies—are using AWS to lower costs, become more agile, and innovate faster.



nClouds is an **AWS Premier Consulting Partner** and award-winning provider of AWS and DevOps consulting and implementation services. We are an integrated team of skilled engineers, architects, developers, project managers, and sales & marketing pros who are passionate about client success, software excellence, and innovation. We enable our clients to deliver better products faster and create awesome customer experiences.



Trusted by INNOVATIVE BRANDS



Kubernetes on AWS: Observability

AGENDA

DETAILS *(All times PT)*

- **11:00 - 11:05 am** - Intro & Session Objectives *by Randy Newell, nClouds*
- **11:05 - 11:20 am** - Observability in Amazon EKS *by Curtis Rissi, AWS*
- **11:20 - 11:35 am** - DIY Observability Using Open Source Tools *by Marius Ducea, nClouds*
- **11:35 - 11:50 am** - Datadog Demo *by Gagan Goswami, nClouds*
- **11:50 - 12:00 noon** - Q&A *with AWS and nClouds*

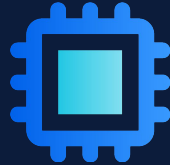
Kubernetes on AWS: Observability

OBJECTIVES



Observability in Amazon EKS

Logging with Amazon CloudWatch, Tracing with AWS X-Ray, Container Insights



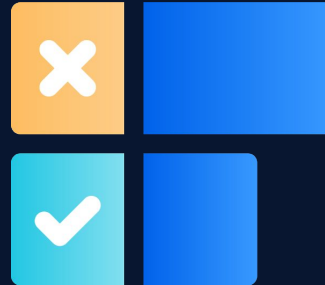
DIY Using Open Source Tools

Monitoring with Prometheus, Grafana, Logging with Elasticsearch/Kibana/Fluent Bit, Tracing with Jaeger



Datadog Demo

Setting Up Datadog Features & Metrics in Amazon EKS



Poll



Observability in Amazon EKS



Curtis Rissi

Senior Solutions Architect



What is **Observability**?

In control theory, **observability** is a measure of how well internal states of a system can be inferred from knowledge of its external outputs.*

Monitoring tells you whether a system is working;
Observability lets you understand why it isn't working.

Good observability allows you to answer the questions you didn't know that you needed to ask.

What is Observability

What: Numeric representation of data measured over intervals of time

Why: Useful for identifying trends, mathematical modeling, and prediction



Metrics



Logs

Observability



Distributed Traces

What: Immutable, timestamped record of discrete events that happened over time

Why: Useful for uncovering emergent and unpredictable behavior

What: Representation of a series of related distributed events that encode the end-to-end request flow through a distributed system

Why: Provides visibility into both the path traversed by a request as well as the structure of a request

AWS Observability Tools

What: Numeric representation of data measured over intervals of time

Why: Useful for identifying trends, mathematical modeling, and prediction



CloudWatch metrics



CloudWatch Logs

AWS observability tools



X-Ray traces

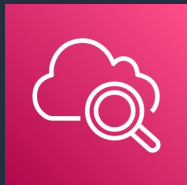
What: Immutable, timestamped record of discrete events that happened over time

Why: Useful for uncovering emergent and unpredictable behavior

What: Representation of a series of related distributed events that encode the end-to-end request flow through a distributed system

Why: Provides visibility into both the path traversed by a request as well as the structure of a request

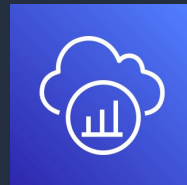
AWS Observability Tools



Amazon CloudWatch Container Insights

Complete visibility of container resources and applications

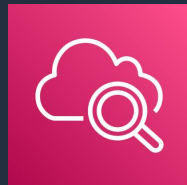
- Monitor applications
- Respond to performance changes
- Optimize resource utilization
- Get a unified view of operational health



AWS X-Ray

Analyze and debug production, distributed applications

- Identify performance bottlenecks
- Troubleshoot root cause
- Trace user requests
- For simple & complex applications



Amazon CloudWatch ServiceLens

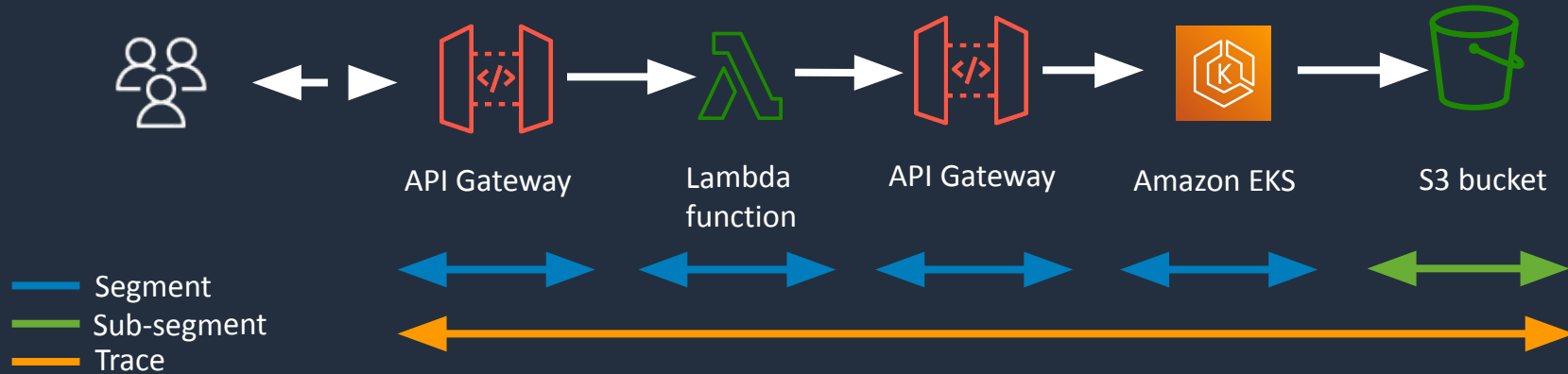
Investigate problems and their effect on your applications

- Integrates CloudWatch and X-Ray
- Pinpoint performance bottlenecks
- Identify impacted users

Distributed Tracing with **AWS X-Ray**

AWS X-Ray: Traces

AWS X-Ray provides an end-to-end, cross-service view of requests made to your application. It gives you an application-centric view of requests flowing through your application by aggregating the data gathered from individual services in your application into a single unit called a trace.



AWS X-Ray: Traces - Continued

Traces individually and in aggregate provide fine grained insights into customer experience and usage of the service/microservices. A few examples might include:

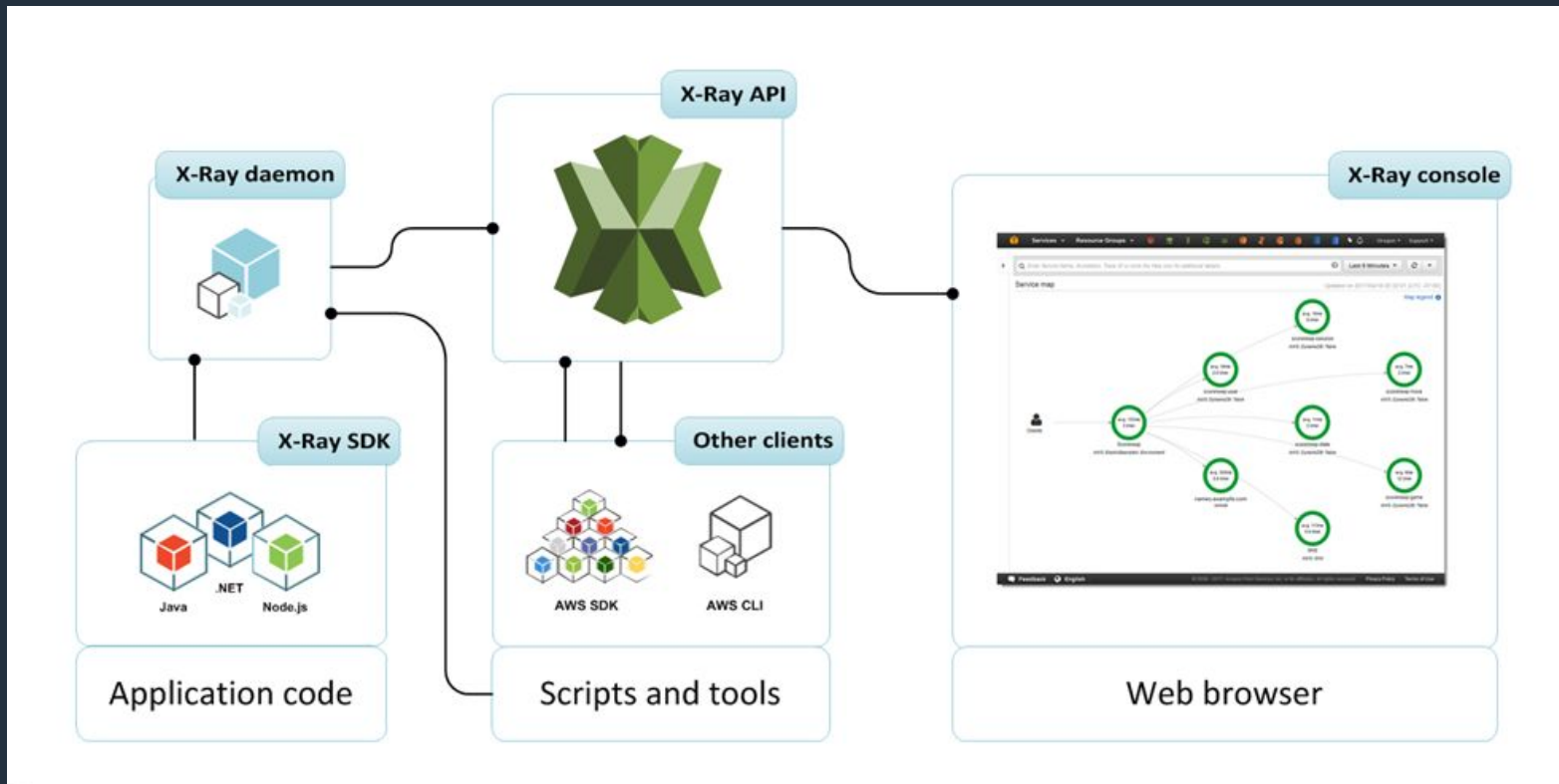
- What was the root cause of a series of API faults?
- Which customers were impacted?
- How many APIs were involved?
- Is my downstream partner meeting their response time SLA?
- How can I tune my critical code paths?
- Are some of our APIs less performant than others? Why?
- What did the stack trace say?

AWS X-Ray: Service Map

AWS X-Ray creates a map of services used by your application with trace data that you can use to drill into specific services or issues.



AWS X-Ray: SDKs and Daemon

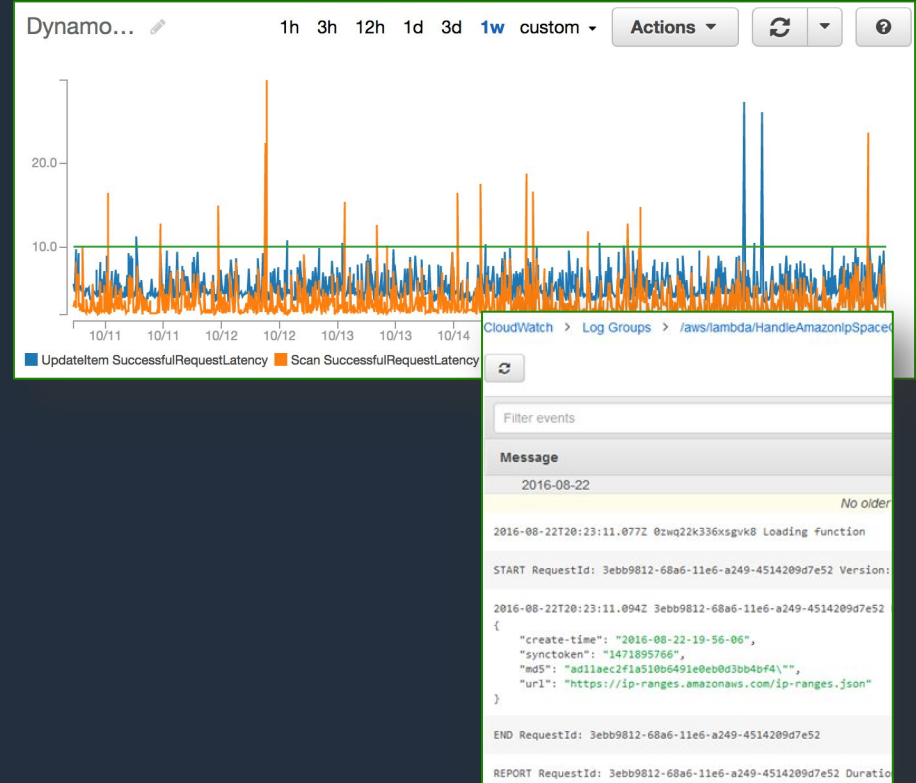


Metrics & Logs

Amazon CloudWatch

Use **AWS-generated metrics, logs, and events** over time to understand the behavior of your system.

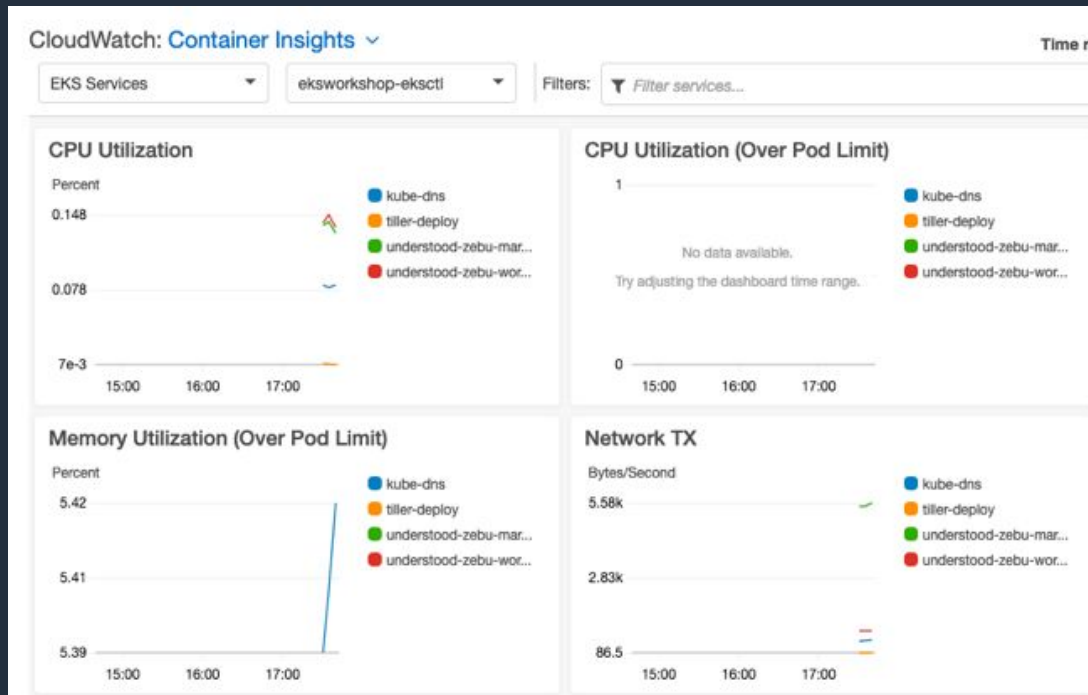
Publish custom metrics, logs, and events for your application specific telemetry.



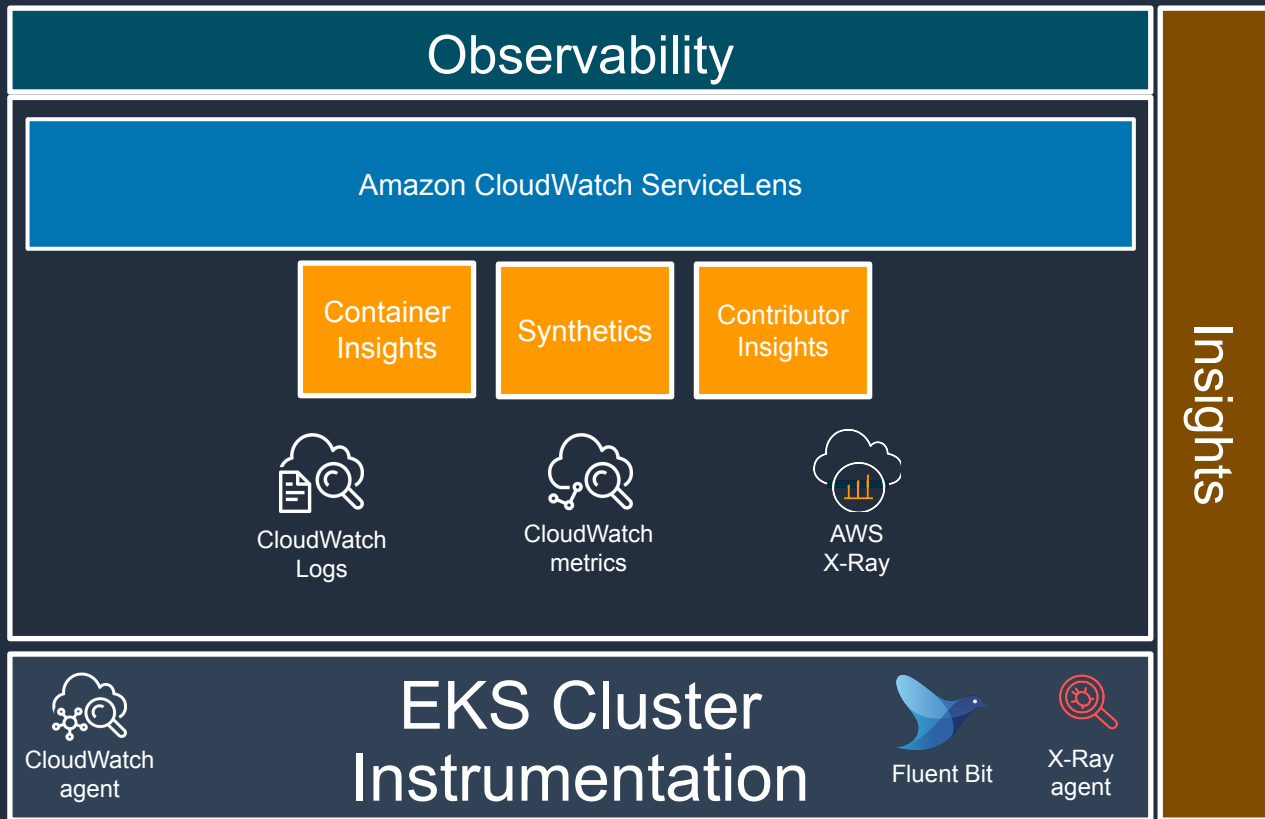
Amazon CloudWatch **Container Insights**

Collect, aggregate and summarize metrics and logs from your containerized apps and microservices.

Provides diagnostic information, such as container restart failures, to help you isolate issues and resolve them quickly.



Pulling it all together...



CloudWatch ServiceLens

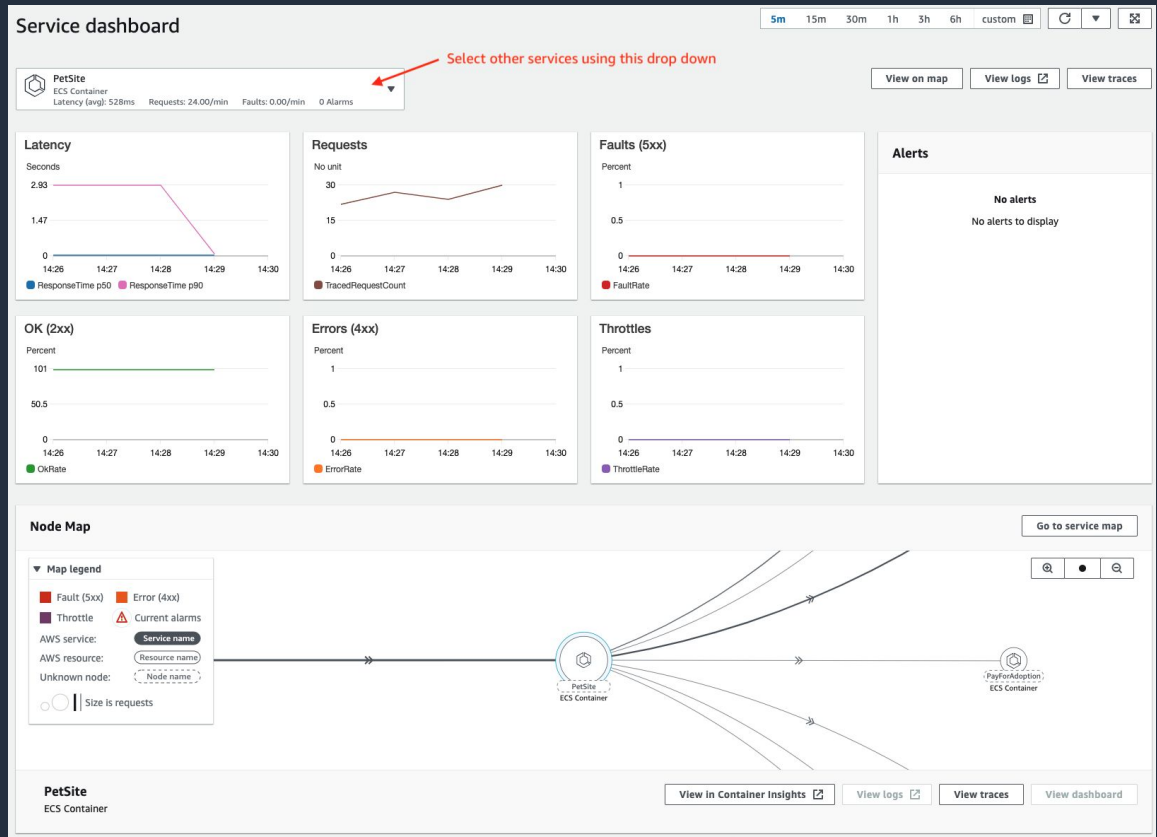
Provides an interactive **Service Map** of your application with trace data that you can use to drill into specific services or issues.

The screenshot shows the AWS CloudWatch ServiceLens interface. At the top, there are navigation options for 'View data for' (account and region) and a search bar for 'Filter by X-Ray group'. A duration selector is set to '5m'. A 'Map legend' on the right lists various error types: Fault (5xx), Error (4xx), Throttle, and Current alarms. The Service Map itself is a network diagram with nodes for 'Client', 'PetSearchAPI', 'PetAdoption', 'PetForAdoption', 'PetSearchAPI', 'PetAdoption', and 'PetForAdoption'. A red circle highlights an error on the 'PetSearchAPI' node, with a red arrow pointing to it from the text 'Visual indication of errors'. Other red arrows point to the 'Filter by X-Ray group' search bar ('Choose an X-Ray group to filter') and the duration selector ('Select the duration you're interested in'). A legend for 'Nodes' and 'Edges' is also present.

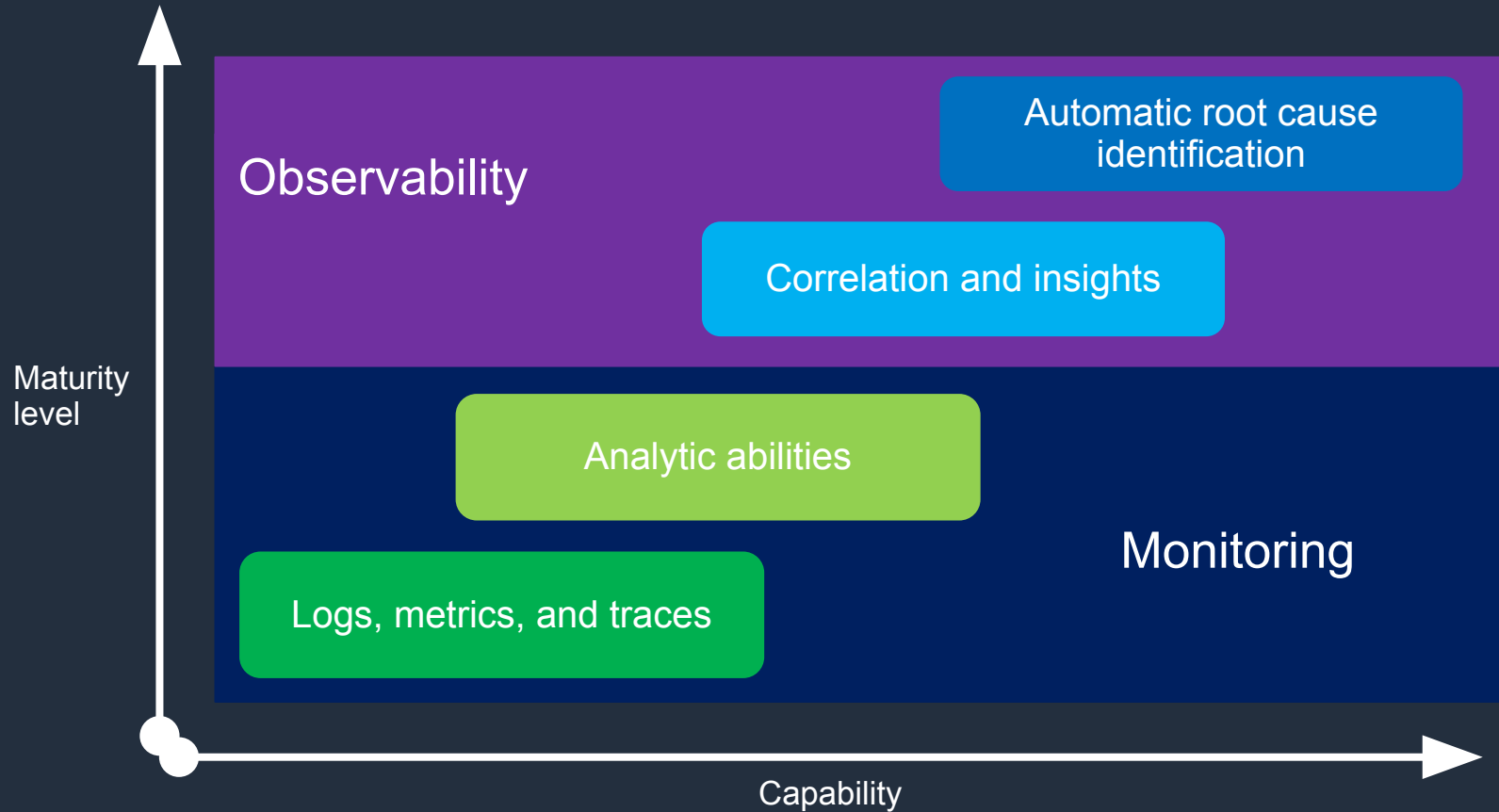


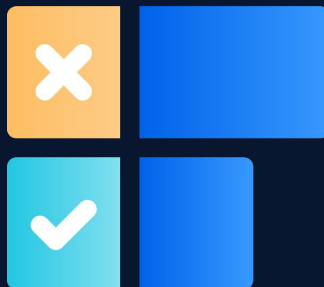
CloudWatch ServiceLens

The **Service Dashboard** shows a quick view of the metrics generated from the service, helping you quickly troubleshoot issues that may be occurring within your applications.



Observability is a Process of Maturity





Poll

DIY Observability Using Open Source Tools on **Amazon EKS**



Marius Ducea

VP of DevOps Practice



DIY Observability with **Open Source tools**

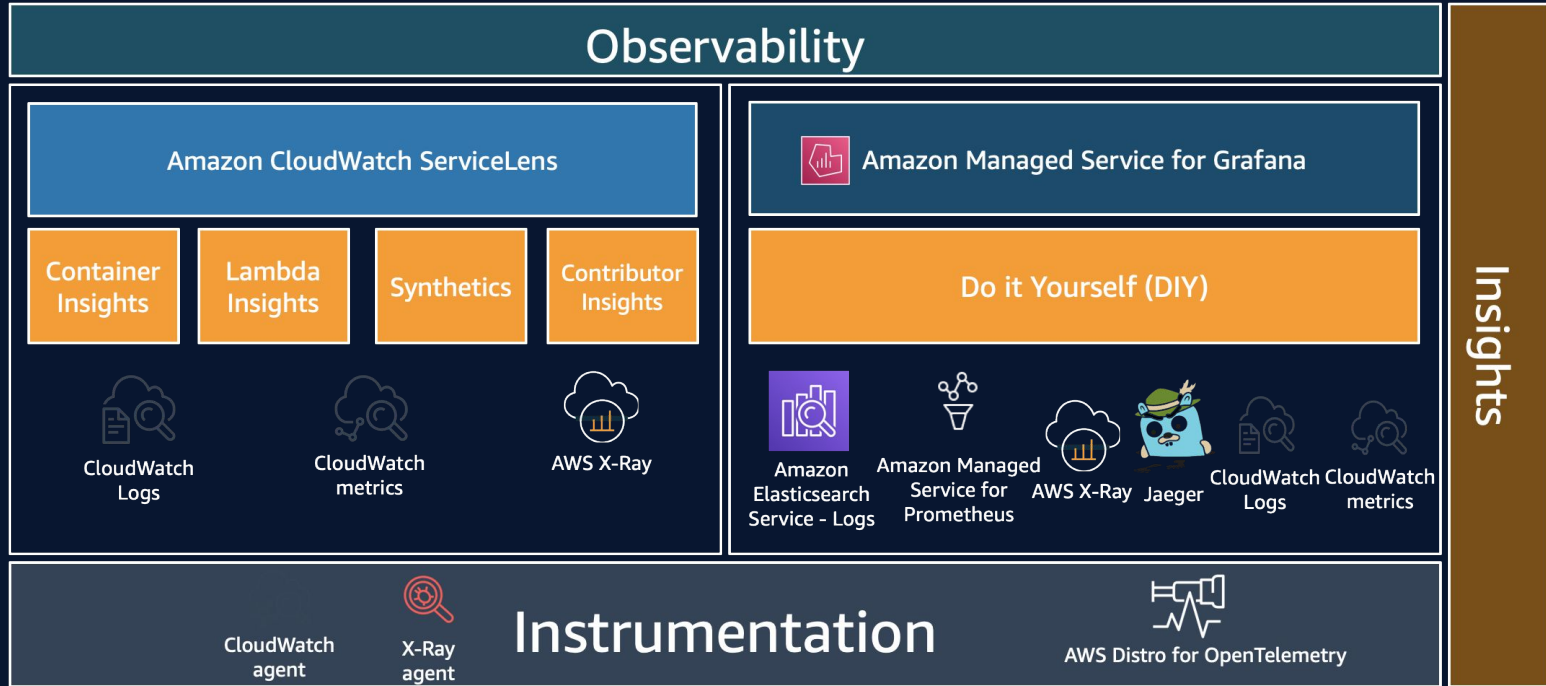
**AWS native tools: CloudWatch Logs and
CloudWatch Logs Insights, X-Ray**

Open Source tools:

- Monitoring: Prometheus/Grafana
- Logging: Elasticsearch/Kibana/Fluent Bit
- Tracing: Jaeger

SaaS solutions: Datadog, Honeycomb, New Relic

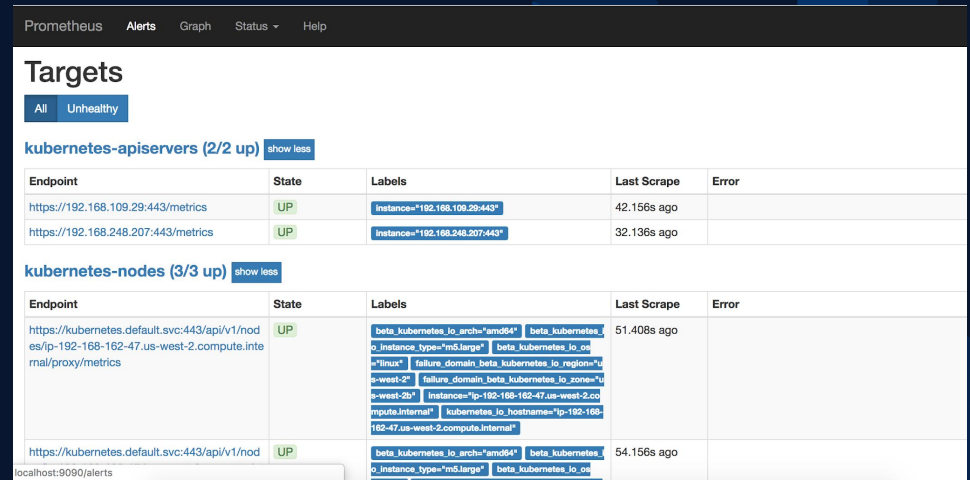
Open source DIY with help from AWS



Monitoring: Prometheus

Prometheus is an open-source systems monitoring and alerting toolkit originally built at SoundCloud.

It was the second project to join the Cloud Native Computing Foundation in 2016, after k8s.



The screenshot shows the Prometheus web interface with the following data:

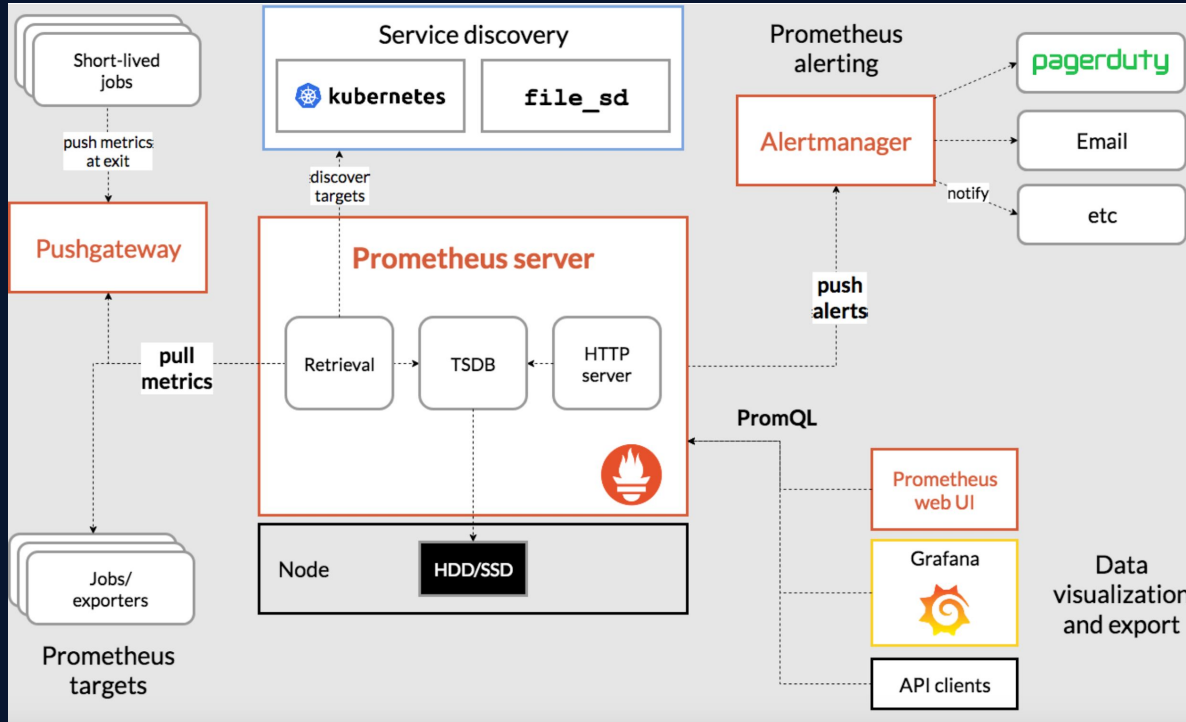
Endpoint	State	Labels	Last Scrape	Error
https://192.168.109.29:443/metrics	UP	instance="192.168.109.29:443"	42.156s ago	
https://192.168.248.207:443/metrics	UP	instance="192.168.248.207:443"	32.136s ago	

Endpoint	State	Labels	Last Scrape	Error
https://kubernetes.default.svc:443/api/v1/nodes/ip-192-168-162-47.us-west-2.compute.internal/proxy/metrics	UP	beta_kubernetes_io_arch="amd64" beta_kubernetes_io_instance_type="m5.large" beta_kubernetes_io_os="linux" failure_domain_beta_kubernetes_io_region="us-west-2" failure_domain_beta_kubernetes_io_zone="us-west-2b" instance="ip-192-168-162-47.us-west-2.compute.internal" kubernetes_io_hostname="ip-192-168-162-47.us-west-2.compute.internal"	51.408s ago	
https://kubernetes.default.svc:443/api/v1/nodes/localhost:9090/alerts	UP	beta_kubernetes_io_arch="amd64" beta_kubernetes_io_instance_type="m5.large" beta_kubernetes_io_os="linux" failure_domain_beta_kubernetes_io_region="us-west-2" failure_domain_beta_kubernetes_io_zone="us-west-2b" instance="ip-192-168-162-47.us-west-2.compute.internal" kubernetes_io_hostname="ip-192-168-162-47.us-west-2.compute.internal"	54.156s ago	

Prometheus features

- A multi-dimensional data model with time series data identified by metric name and key/value pairs.
- **PromQL**, a flexible query language to leverage this dimensionality.
- Time series collection happens via a **pull model** over HTTP.
- Targets are discovered via service discovery or static configuration.

Prometheus architecture



Prometheus installation on EKS

```
kubectl create namespace prometheus
```

```
helm install prometheus prometheus-community/prometheus \  
  --namespace prometheus \  
  --set alertmanager.persistentVolume.storageClass="io1" \  
  --set server.persistentVolume.storageClass="io1"
```

```
kubectl port-forward -n prometheus deploy/prometheus-server 8080:9090
```

Amazon Managed Service for Prometheus (AMP) * preview



A serverless Prometheus-compatible monitoring service for metrics to securely monitor container environments at scale

Fully managed, secure, and highly available using multi-AZ deployments

Use the same open source Prometheus data model and query language as they do today to monitor the performance of their containerized workloads

No up-front investments required to use the service, and customers only pay for the number of metrics ingested

Improved scalability, availability, and security without having to manage the underlying infrastructure

Monitoring: Grafana

Uniquely visualizes and combines insights from multiple open source, cloud, and third-party data sources without moving the data.

Has become among the most popular operational dashboard technologies in the world.



Grafana features

- Visualization
 - Dynamic dashboards, explore metrics and logs
- Alerting
 - Define rules and send alerts to PagerDuty, Opsgenie, Slack
- 30+ Datasources: AWS native & many others
 - Prometheus, Elasticsearch, InfluxDB, CloudWatch, etc.
- Many dashboards and plugins

Grafana installation on EKS



```
mkdir ${HOME}/environment/grafana

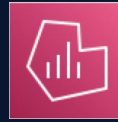
cat << EoF > ${HOME}/environment/grafana/grafana.yaml
datasources:
  datasources.yaml:
    apiVersion: 1
    datasources:
      - name: Prometheus
        type: prometheus
        url: http://prometheus-server.prometheus.svc.cluster.local
        access: proxy
        isDefault: true
EoF
```

```
kubectl create namespace grafana

helm install grafana grafana/grafana \
  --namespace grafana \
  --set persistence.storageClassName="gp2" \
  --set persistence.enabled=true \
  --set adminPassword='EKS!sAWSome' \
  --values ${HOME}/environment/grafana/grafana.yaml \
  --set service.type=LoadBalancer
```



Amazon Managed Service for **Grafana** (AMG) * preview



Scalable, secure, and highly available fully-managed Grafana service.

Automatic scaling, helps offload operational management

Analyze, monitor, and alarm across multiple data sources; native AWS as well as 3rd party

Native integration with multiple AWS Services for Enterprise-ready security

Easily upgrade to Grafana Enterprise from AWS marketplace

Simple pay-as-you-go AWS billing

Using the Prometheus operator

Using the CoreOS Prometheus Operator, we get configurations for retrieving Kubernetes-native metrics, Grafana installation with preconfigured Prometheus dashboards, AlertManager deployment, Elasticsearch, and many other useful tools all right out of the box...

```
helm repo add stable https://kubernetes-charts.storage.googleapis.com/
helm install prom-operator stable/prometheus-operator --namespace=monitoring

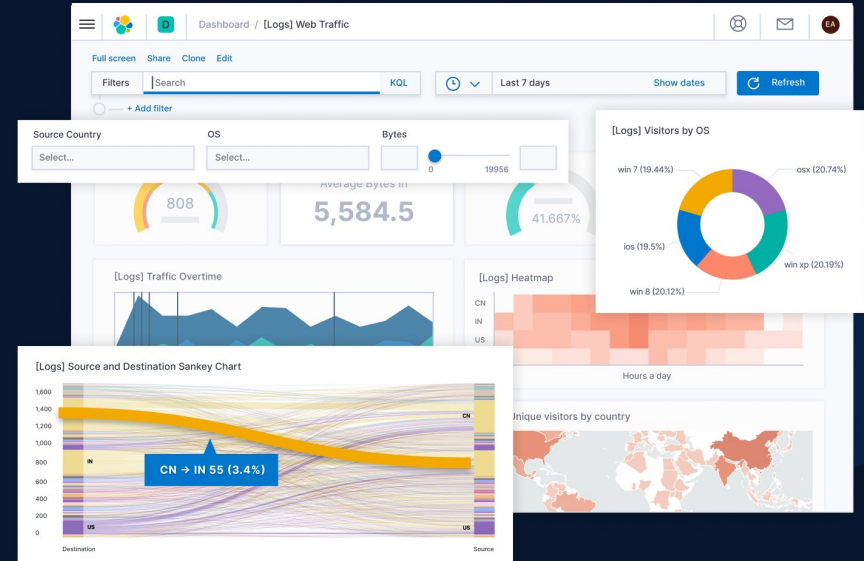
$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
alertmanager-prom-operator-prometheus-o-alertmanager-0  2/2     Running   2           5h30m
elasticsearch-6f4c587475-sdpc2      1/1     Running   1           22h
prom-operator-grafana-684965779-2fjhr  2/2     Running   2           5h31m
prom-operator-kube-state-metrics-756d8d6849-n949p      1/1     Running   1           5h31m
prom-operator-prometheus-node-exporter-66z2m          1/1     Running   1           5h31m
prom-operator-prometheus-o-operator-847cb84c6d-mqsnv  2/2     Running   2           5h31m
prometheus-prom-operator-prometheus-o-prometheus-0    3/3     Running   4           5h30m

kubectl port-forward -n monitoring prometheus-prom-operator-prometheus-o-prometheus-0 9090
```


Logging: Elasticsearch / Kibana

Elasticsearch: the fast and scalable search engine at the heart of the Elastic Stack.

Kibana: the powerful and customizable visualization layer and end user interface of the ELK Stack.



Logging: Fluent Bit

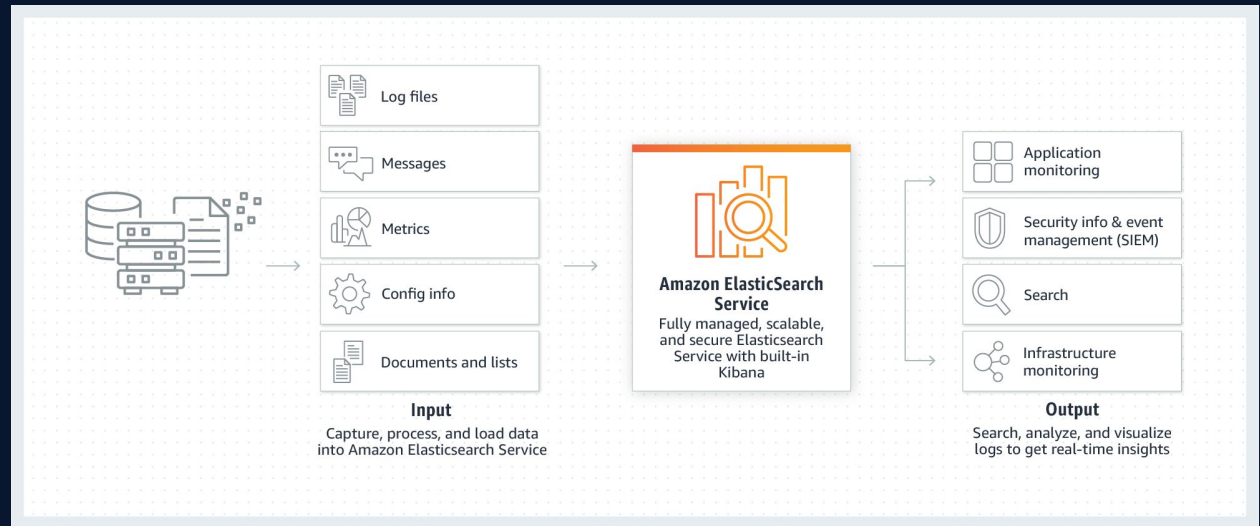
Fluent Bit is a lightweight and extensible **Log Processor** that comes with full support for Kubernetes:

- Process Kubernetes containers logs from the file system or Systemd/Journald.
- Enrich logs with Kubernetes Metadata.
- Centralize your logs in third party storage services like Elasticsearch, InfluxDB, HTTP, etc.

Amazon Elasticsearch Service

Fully managed, scalable, and secure Elasticsearch service

- Easy to deploy and manage
- Highly scalable and available
- Highly secure
- Cost-effective



Fluent Bit installation on EKS

Fluent Bit is usually deployed as a DaemonSet, so it will be available on every node of your Kubernetes cluster.

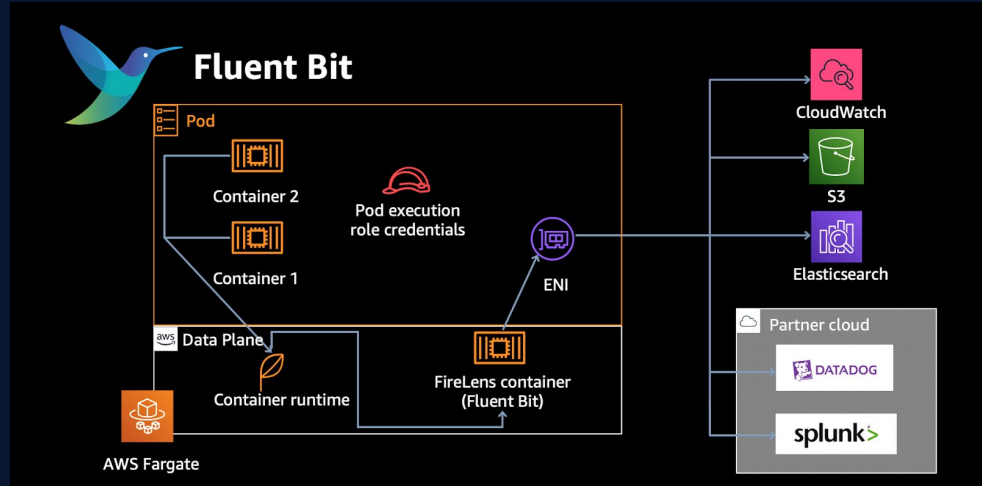
```
helm repo add fluent https://fluent.github.io/helm-charts
helm install fluent-bit fluent/fluent-bit
```

Fluent Bit for EKS on Fargate



With EKS on Fargate, each pod acts as an isolated Kubernetes node so the only way to get logs was to run a sidecar along with your application.

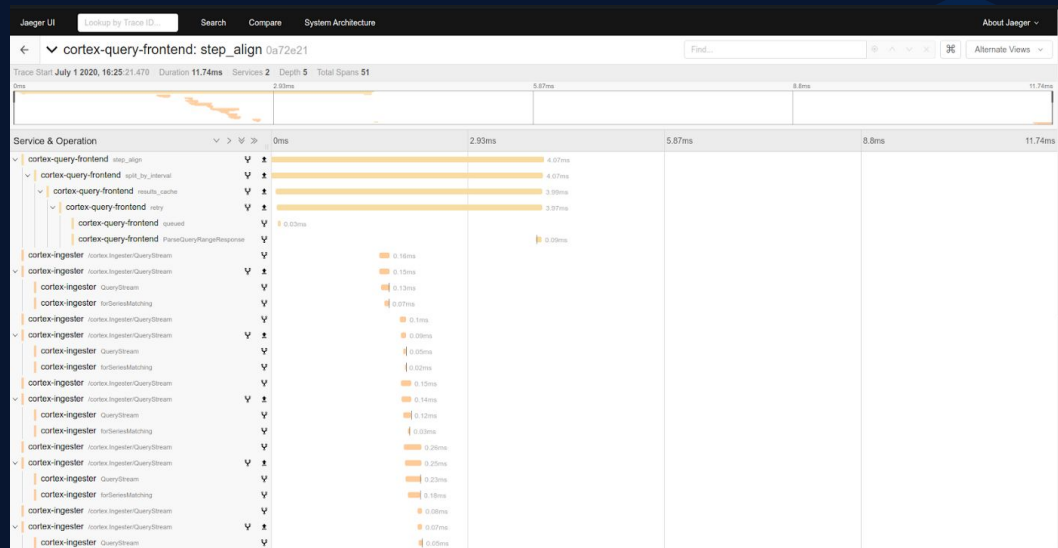
Fluent Bit for Amazon EKS on AWS Fargate allows you to define a k8s ConfigMap and namespace (aws-observability) for pods you want to have an automatic “hidecar” attached to your pods.



Tracing: Jaeger



Jaeger is an open source, end-to-end distributed tracing system used for monitoring and troubleshooting microservices-based distributed systems.



Jaeger features



- High Scalability
 - No single points of failure and scales with the business needs
- Multiple storage backends
 - Cassandra 3.4+ and Elasticsearch 5.x/6.x/7.x.
- Native support for OpenTracing
 - Allows developers to add performance-monitoring features to their projects using a common API specification that is non-vendor-specific
- All Jaeger backend components expose Prometheus metrics by default

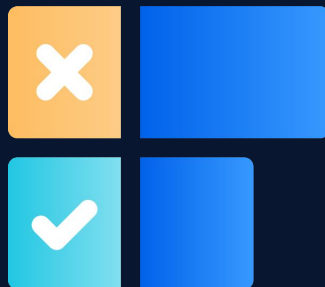
Jaeger installation



Jaeger backend is distributed as a collection of Docker images.

Deployment to Kubernetes clusters is assisted by a Kubernetes operator, Kubernetes templates, or a Helm chart.

Jaeger Agent installed as a DaemonSet or Sidecar.



Poll

Demo: Datadog



Gagan Goswami

DevOps Engineer



What is Datadog?

Datadog is a dedicated monitoring service that gives a complete picture of EKS cluster's health and performance. Datadog **centralizes** all these sources of data into a **single platform**. Datadog agent collects and forwards **metrics, logs and traces** from each node and container running on nodes.



Datadog Kubernetes **Agents**



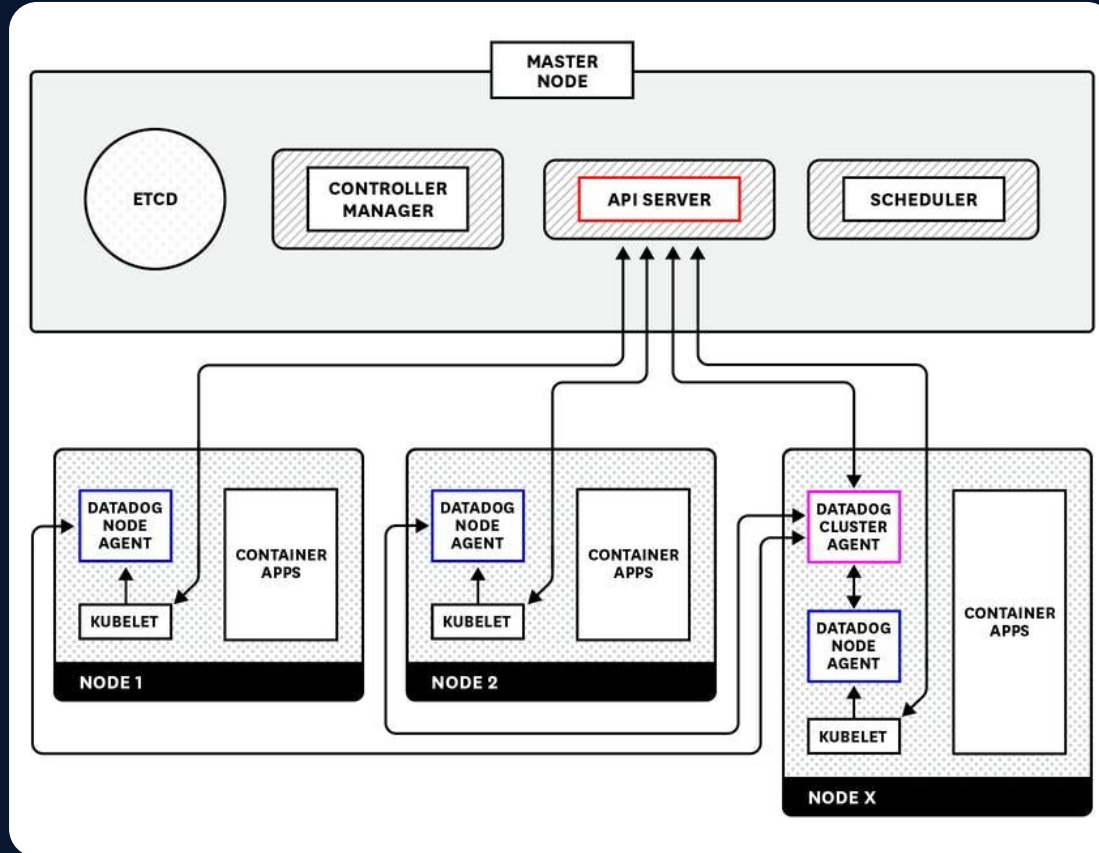
Cluster agent

Communicates with Kubernetes API servers to collect **cluster-level** information. It runs on a single node and serves as a proxy between API servers and **node-agents**.

Node-based agent

Reports data from each node's **kubelet**. It works as **daemonset** that runs one single agent per node within cluster.

K8s Datadog Agent Architecture



Agent Provisioning



Helm Chart



Kubernetes
Daemonset



Kubernetes
Operator

AWS Integration



Account: **New Account**

Role Delegation

Access Keys (GovCloud or China Only)

Choose a method for setting up the necessary AWS role (we recommend using CloudFormation).

If using the automatic setup, complete the CloudFormation launch process in AWS, and then insert the AWS account ID and generated role name back in this form when completed.

Automatically Using CloudFormation

Manually

Remove Account



Demo

Special OFFERS



**Free Observability
Assessment**
for all eligible attendees





Q&A

Kubernetes on AWS: Observability

PRESENTERS



Curtis Rissi

Senior Solutions Architect



Marius Ducea

VP of DevOps Practice



Gagan Goswami

DevOps Engineer

