Observability of Microservices

USING OPEN SOURCE TOOLS

Conf42 DevOps 2023, Shubham Srivastava

Hey, I’m Shubham 👋

Founding Developer Relations Engineer at Zenduty - an advanced incident management and response orchestration platform.

Expert at making mistakes, learning from them and advocating for best practices for orgs setting up their DevOps, SRE and Production Engineering teams.

Shhhhhubham **/shubham67**

What is this talk about?

-Quick **brief** on observability

-Why we need **observability from Day 1**

-Setting up an observability system **using open source tools** *(speedrun)*

-Got observability set up, **what next?**

What is **observability**?

Observability is the ability to measure the internal states of a system by examining its outputs.

Observability gives engineers a proactive approach to optimizing their systems. It provides a connected real-time view of all the operational data in your software system, as well as the flexibility to **ask questions** on the fly about your applications and infrastructure to get the answers you need.

Why do I keep hearing about it all the time **now**?

Simply put - Software complexity is increasing at an exponential rate; products are innovating in a crazy unpredictable direction.

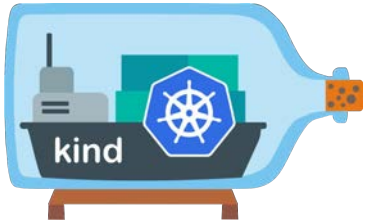
Why do we need **Day 1 Observability?**

-Failures occur at **all stages** of software development.

-We're often in the **dark about their occurrence** and have little clues to begin a diagnosis.

-They lead to **business impact** sooner or later.

The Meat - Setting It All Up



kind

To run a local Kubernetes cluster with Docker container nodes.

Prerequisites

If you have go(1.17+) and docker installed -

```
go install sigs.k8s.io/kind@v0.17.0 &&
kind create cluster
```

kubectl

Prerequisites

To run commands against Kubernetes clusters

```
brew install kubectl
```

helm

Prerequisites

To install Odigos and the observability backend using helm charts.

```
brew install helm
```

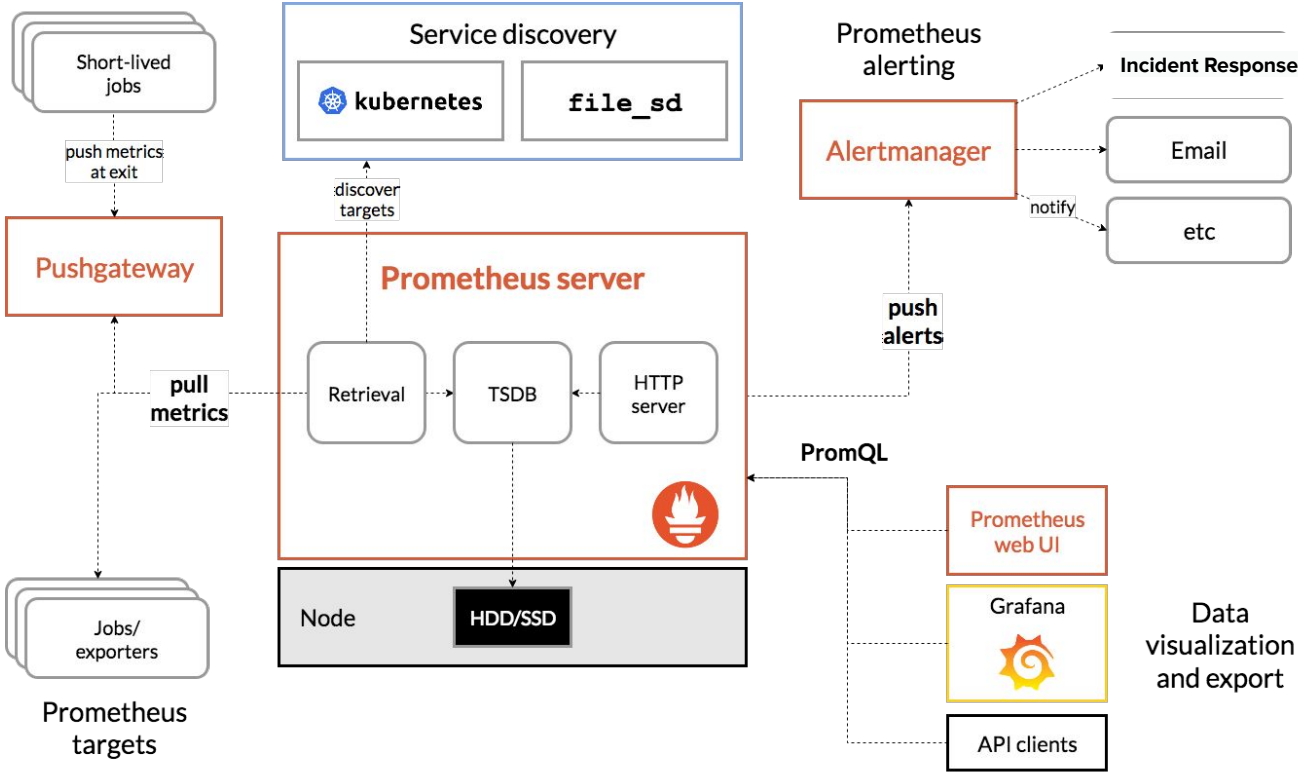
Observability Backend - To Store and Analyse Observability Data

For dashboards and visualization of the data

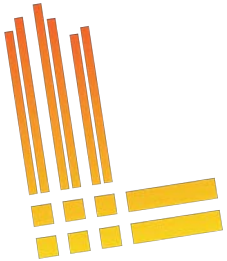
Grafana

To Store Metric Data

Prometheus



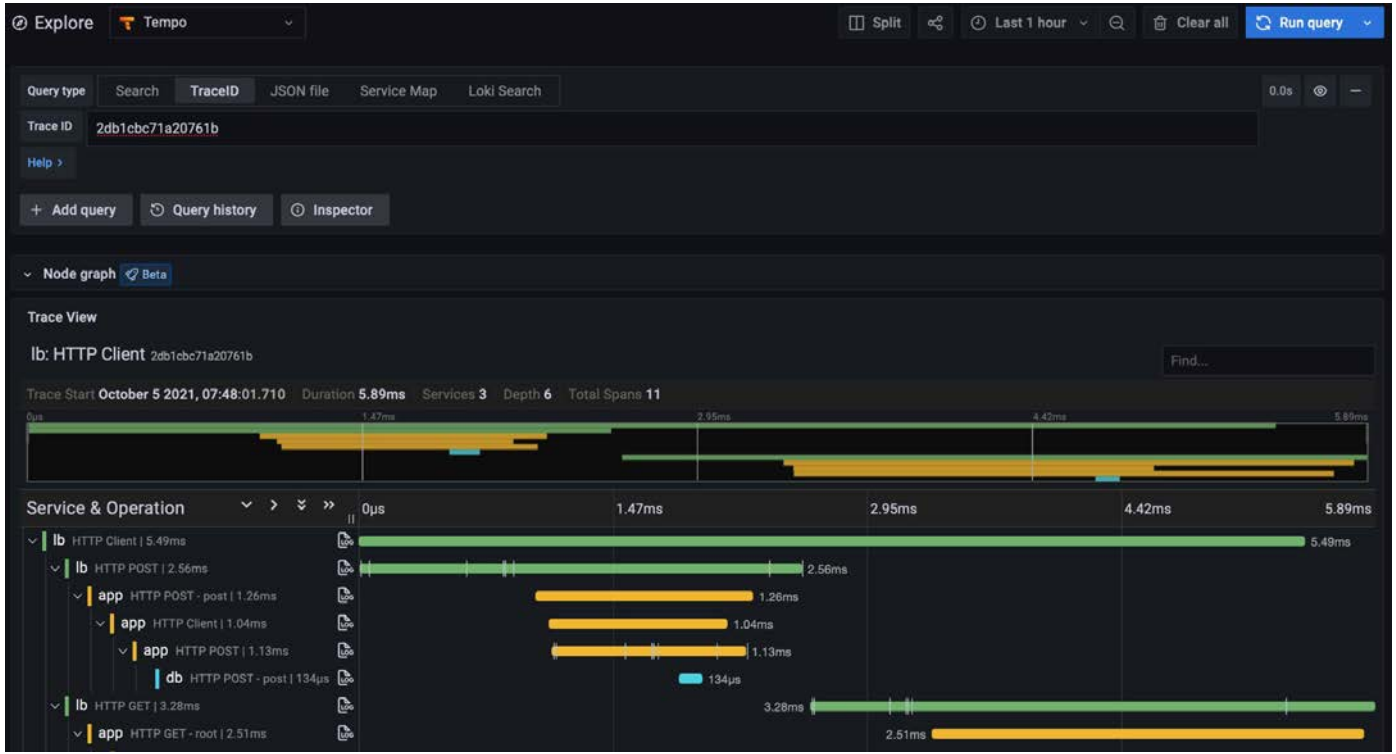
To Store Log Data



Loki

zenduty

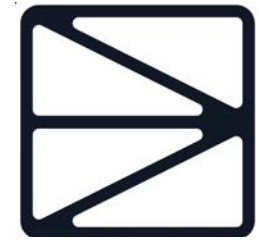
To Store Distributed Tracing Data



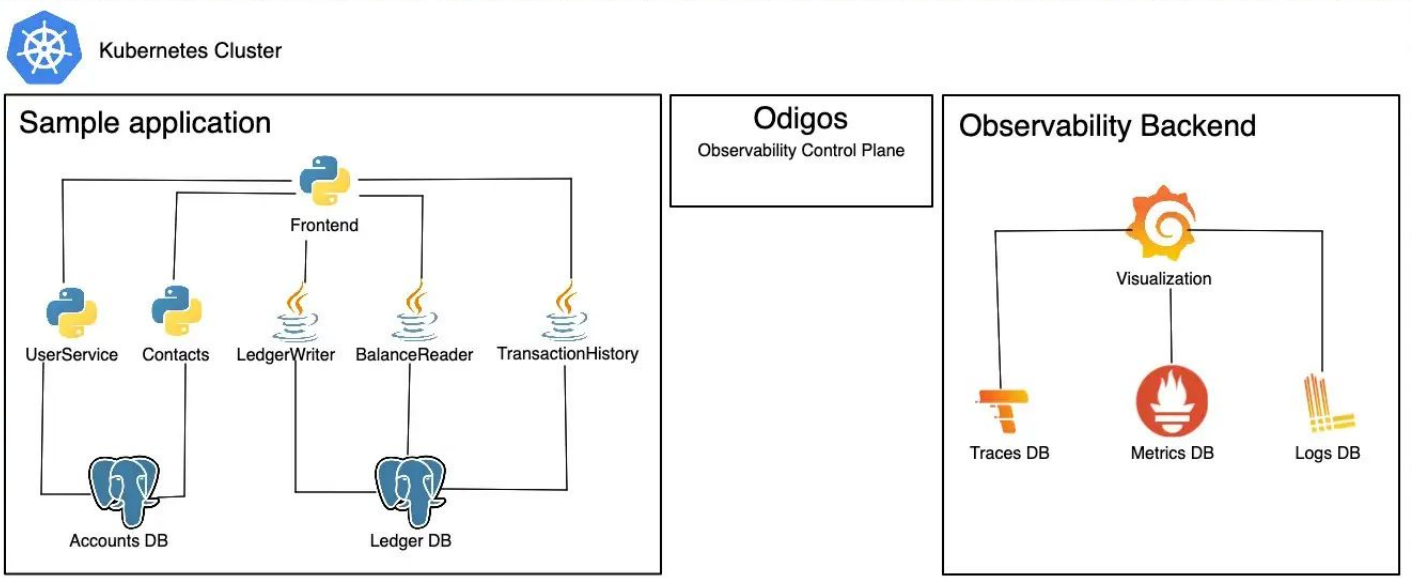
zenduty

Observability Control Plane- To automatically instrument our applications

For (automatic extraction of traces, metrics, and logs), collectors deployment, and configuration.



Odigos



Target Application

A microservices-based application written in Java/Python.

We'll be using a fork of the bank-of-anthos example application made by Google

Target Application

Deploy the application -

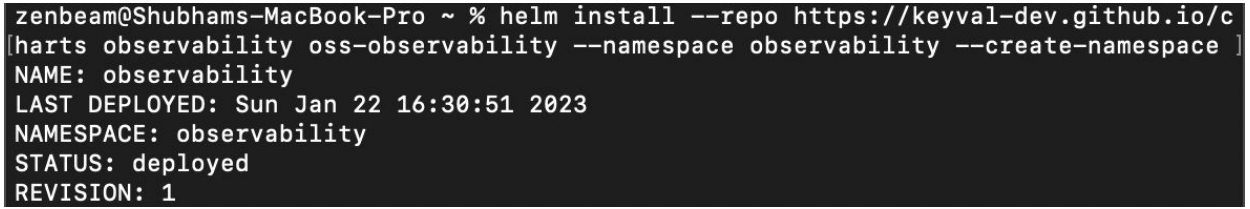
```
kubectl apply -f
https://raw.githubusercontent.com/keyval-dev/
bank-of-athnos/main/release/kubernetes-manife
sts.yaml
```

```
zenbeam@Shubhams-MacBook-Pro ~ % kubectl apply -f https://raw.githubusercontent.com/keyval-dev/
bank-of-athnos/main/release/kubernetes-manifests.yaml
secret/jwt-key created
configmap/environment-config created
configmap/service-api-config created
configmap/demo-data-config created
statefulset.apps/accounts-db created
service/accounts-db created
configmap/accounts-db-config created
statefulset.apps/ledger-db created
configmap/ledger-db-config created
service/ledger-db created
deployment.apps/balancereader created
service/balancereader created
deployment.apps/contacts created
service/contacts created
deployment.apps/ledgerwriter created
service/ledgerwriter created
deployment.apps/transactionhistory created
service/transactionhistory created
deployment.apps/userservice created
service/userservice created
deployment.apps/frontend created
service/frontend created
deployment.apps/loadgenerator created
```

Installing Observability Backend

Install the helm chart by executing -

```
helm install --repo
https://keyval-dev.github.io/charts
observability oss-observability --namespace
observability --create-namespace
```



Connecting it all with Odigos

Install Odigos via the helm chart -

```
helm repo add odigos
https://keyval-dev.github.io/odigos-charts/
helm install my-odigos odigos/odigos
--namespace odigos-system --create-namespace
```

```
zenbeam@Shubhams-MacBook-Pro ~ % helm repo add odigos https://keyval-dev.github.io/odigos
-charts/
helm install my-odigos odigos/odigos --namespace odigos-system --create-namespace
"odigos" has been added to your repositories
NAME: my-odigos
LAST DEPLOYED: Sun Jan 22 16:31:31 2023
NAMESPACE: odigos-system
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Odigos installed successfully. For getting started go to the UI by running:

kubectl port-forward svc/odigos-ui 3000:3000 -n odigos-system

Then, go to: http://localhost:3000
```

```
kubectl port-forward svc/odigos-ui 3000:3000
-n odigos-system
```

Connecting Grafana

Port forward to your Grafana instance by:

```
kubectl port-forward
svc/observability-grafana -n observability
3000:80
```

Navigate to http://localhost:3000

1)Enter admin as the username

2)For the password enter the output of the following command:

```
kubectl get secret -n observability
observability-grafana -o
jsonpath="{.data.admin-password}" | base64
--decode
```

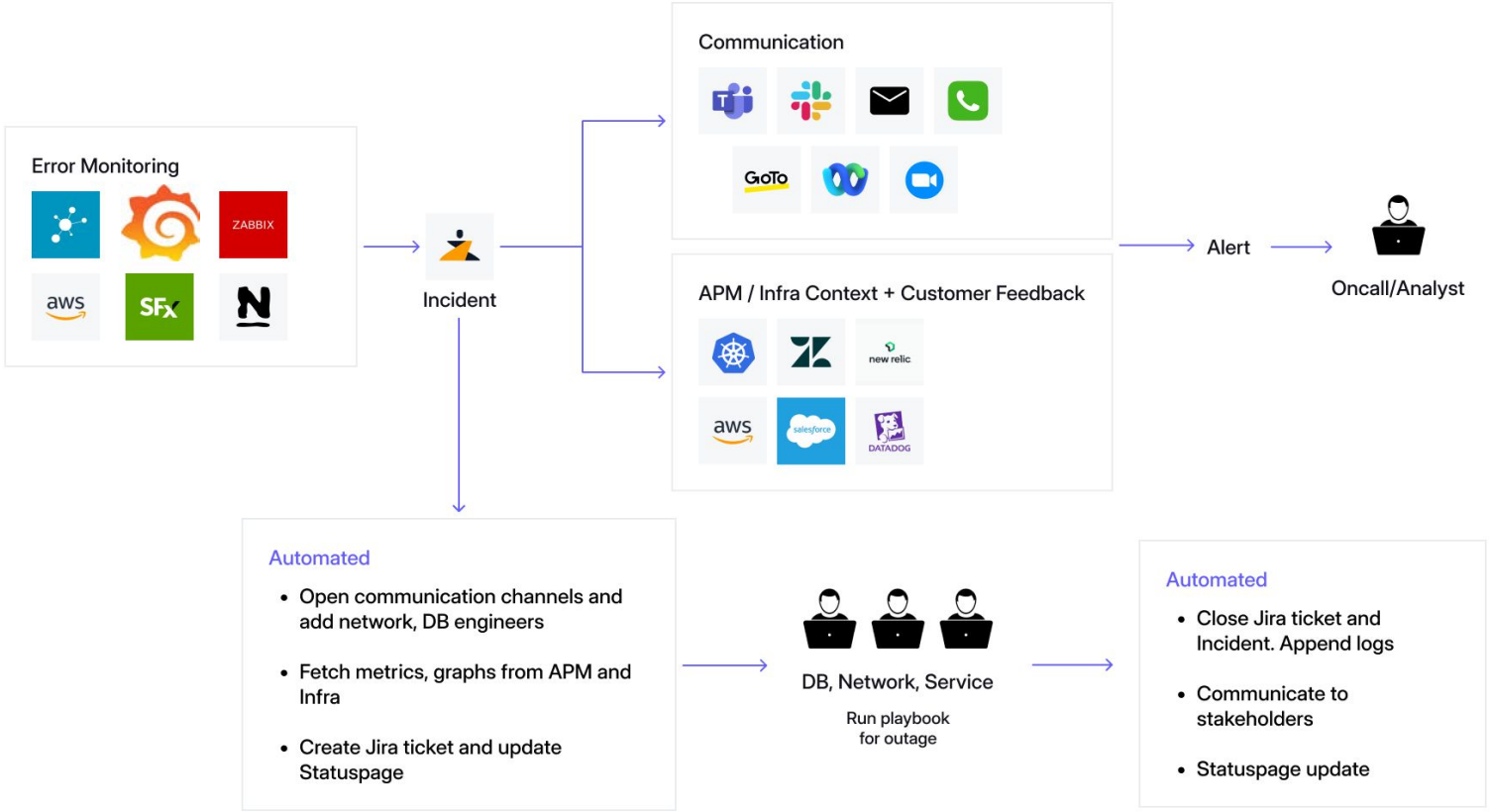
Time to See the Power of Data

We’ve learnt how easy it can be to extract and ship logs, traces, and metrics using only open-source solutions. In addition, we were also able to generate traces, metrics, and logs from an application **within minutes.**

Great! We will now be able to detect and have all the data to diagnose and fix issues.

But how do we approach incident management here for quicker resolution ?

Getting the best out of your Observability data during incidents



zenduty

zenduty

Thanks for listening.

Questions ?


@Shhhhubham


/shubham67


community.zenduty.com

Resources: bit.ly/o11yOSS