Google Summer of Code 2022

Project Proposal

KubeBot

Basic Details:

- Name: Priyanshu Raj Shrivastava
- Email: priyanshuraj400@gmail.com
- GitHub: priyanshuraj400
- LinkedIn: priyanshuraj400
- Contact: +91 8602500319

Project Introduction:

A. Problem

Kubernetes is a great tool for container orchestration, running Kubernetes in your production environment is getting traction in the Cloud industry. With growing DevOps tools, it is now a tedious and time-consuming task SREs and developers to continuously monitor their remote applications running inside a multicluster Kubernetes environment. Though there are some great tools using which we can monitor Kubernetes tools there is a need for easy deployment setup and an alerting tool that people can use to get alerts when something goes wrong inside their application running in a Kubernetes environment. It is often time-consuming to figure out the root cause of such issues, and most importantly not all alerts and issues are important and do not need human interaction. This brings us to address the problem of out of box metrics, traces, events and logs collection for

applications running inside Kubernetes, reporting all the collected data on a single dashboard and push notifications to users for critical ones. The collected metrics, traces, events and logs will help to debug the application running inside Kubernetes faster and effectively.

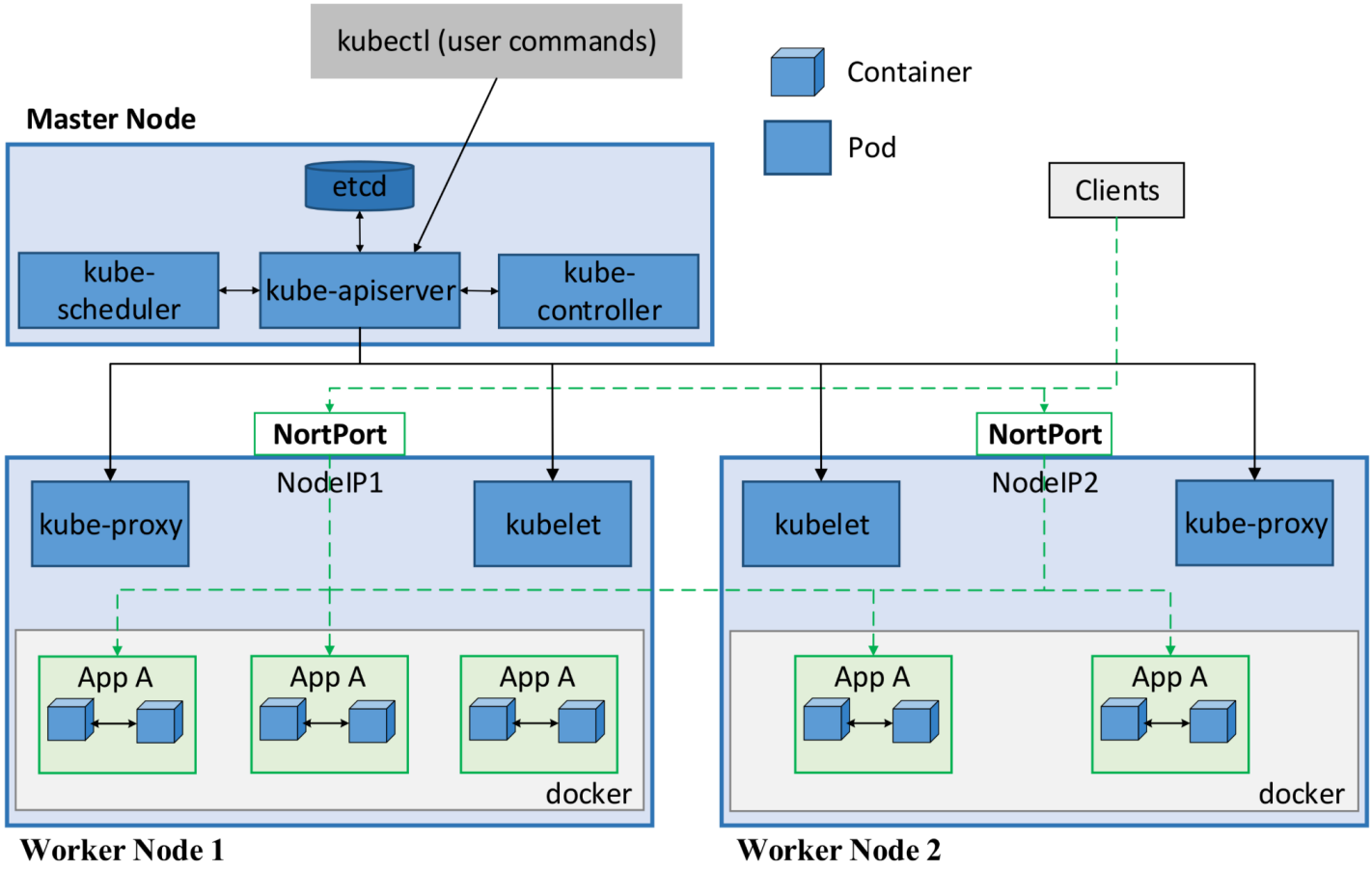

Fig. - High level architecture of Kubernetes, API server will be used to get Metrics, Traces, Events and Logs

B. Proposed Architecture (including pipeline and bot)

There are two components to monitor:
1) Kubernetes Cluster
2) Applications Running inside Kubernetes

1) Kubernetes Cluster

To Monitor Kubernetes Cluster, we can use the exposed Kubernetes metrics as mentioned in the official kubernetes documentation https://kubernetes.io/docs/concepts/cluster-administration/system-metrics/ to an OpenTelemetry Collector. A metrics collector tool like Prometheus can be used to collectively collect all the Kubernetes metrics and send them to OpenTelemetry collector for common ingestion of collected metrics.
In a similar fashion, Kubernetes events and logs

(https://kubernetes.io/docs/concepts/cluster-administration/logging/) will be sent to a common OpenTelemetry collector.
To monitor Kubernetes Node health (https://kubernetes.io/docs/tasks/debug-application-cluster/monitor-node-health/) Prometheus will be used in conjunction with Grafana to see the Node health.

2) Applications Running inside Kubernetes

To monitor customer applications, we can enable the end to end monitoring if the user applications are using any of the opensource monitoring tools within their applications like Apache SkyWalking, Metrics, Datadog, Hyper trace, Newrelic, Honeycomb or OpenTelemetry SDKs. The user can enable the out of the box instrumentation of their application using such tools and Kubebot will provide a seamless experience using docker and sidecar injection to achieve seamless monitoring.

The end to end flow and pipeline for the Kubernetes monitoring involves:

- Applications having monitoring tools installed with them extracting metrics, events, logs, and traces
- Export all the collected metrics, events, logs, and traces to the outer world through Kubernetes APIs
- Kubebot pulls metrics using exposed Kubernetes APIs by internally using Prometheus
- Kubebot also collects logs from deployed fluent bit
- All the collected data will be sent to a common collector in the backend of Kubebot. The collector internally uses the OpenTelemetry collector
- OpenTelemetry collector then send the structured telemetry to common storage and Graphana dashboard for visualization
- The telemetry data stored in the database will be used to train the Machine Learning model inside kubebot for better alerting
- Initially all the alerts will be sent to Grafana and user can select the important alerts, by using reinforcement and support vector machine, the Kubebot will be trained to give better alerts over time

It is essential to utilise the opensource tools available to build a reliable and scalable solution, let's discuss in details each of the open-source tools which will be used in building KubeBot

Prometheus

Prometheus is a free software application used for event monitoring and alerting. It records real-time metrics in a time series database built using a HTTP pull model, with flexible queries and real-time alerting. In Kubebot, Prometheus will help us export all the metrics to Grafana and OpenTelemetry collector along with alerts to apps like Slack etc.

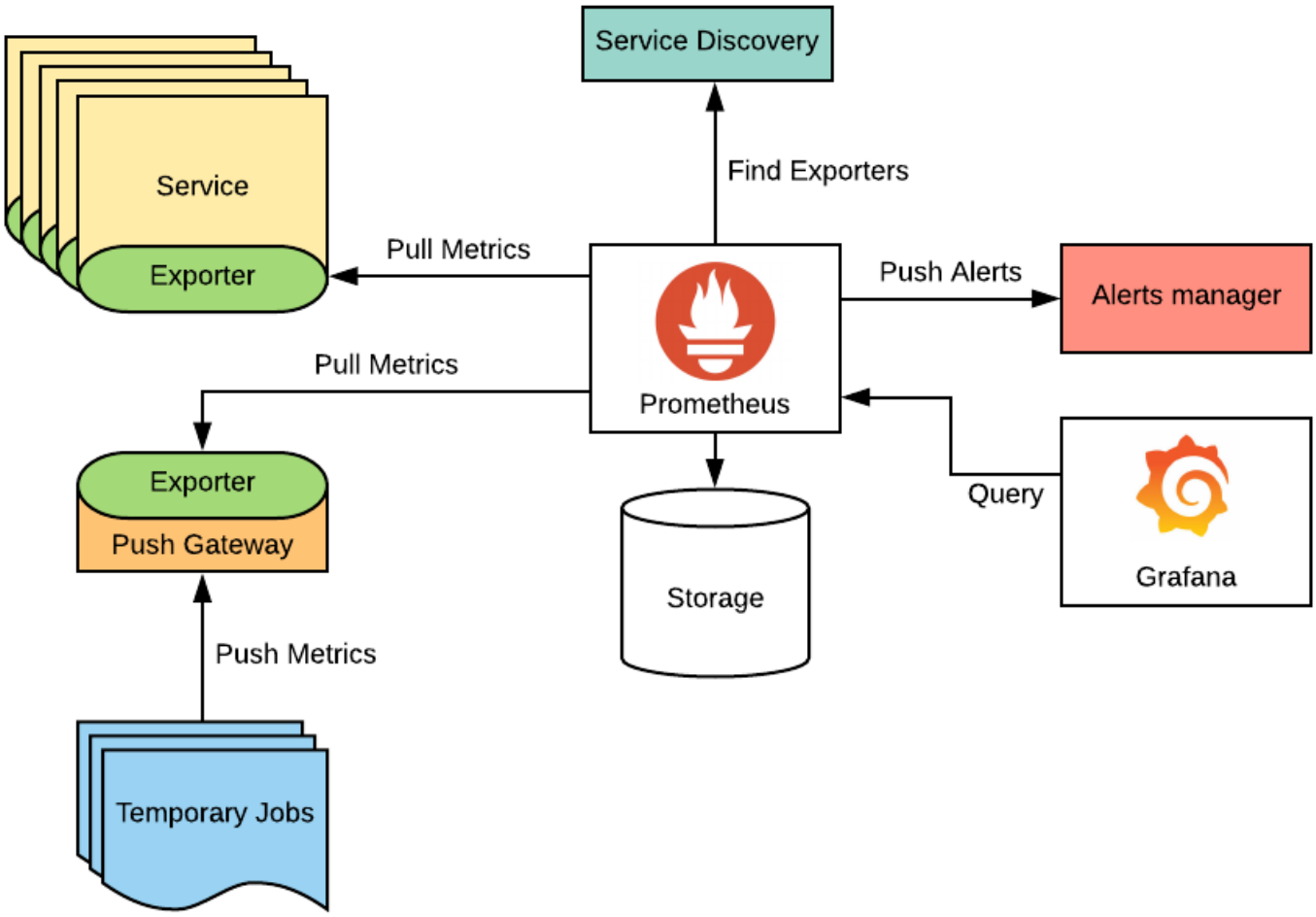

Fig. Generic flow using Prometheus

As shown in the figure above, the Prometheus is used to collect all the Metrics from any environment and infrastructure in which user applications are running and is used to send the metrics to the Grafana dashboard. It also provides the support for alert managers and sends the data to alert a manager to notify the user over slack or any of their prefered applications.

The same model is used to architect the alerting in the Kubernetes environment, the details end to end flow when using Prometheus in conjunction with Graphana, alert manager and OpenTelemetry collector is shown below

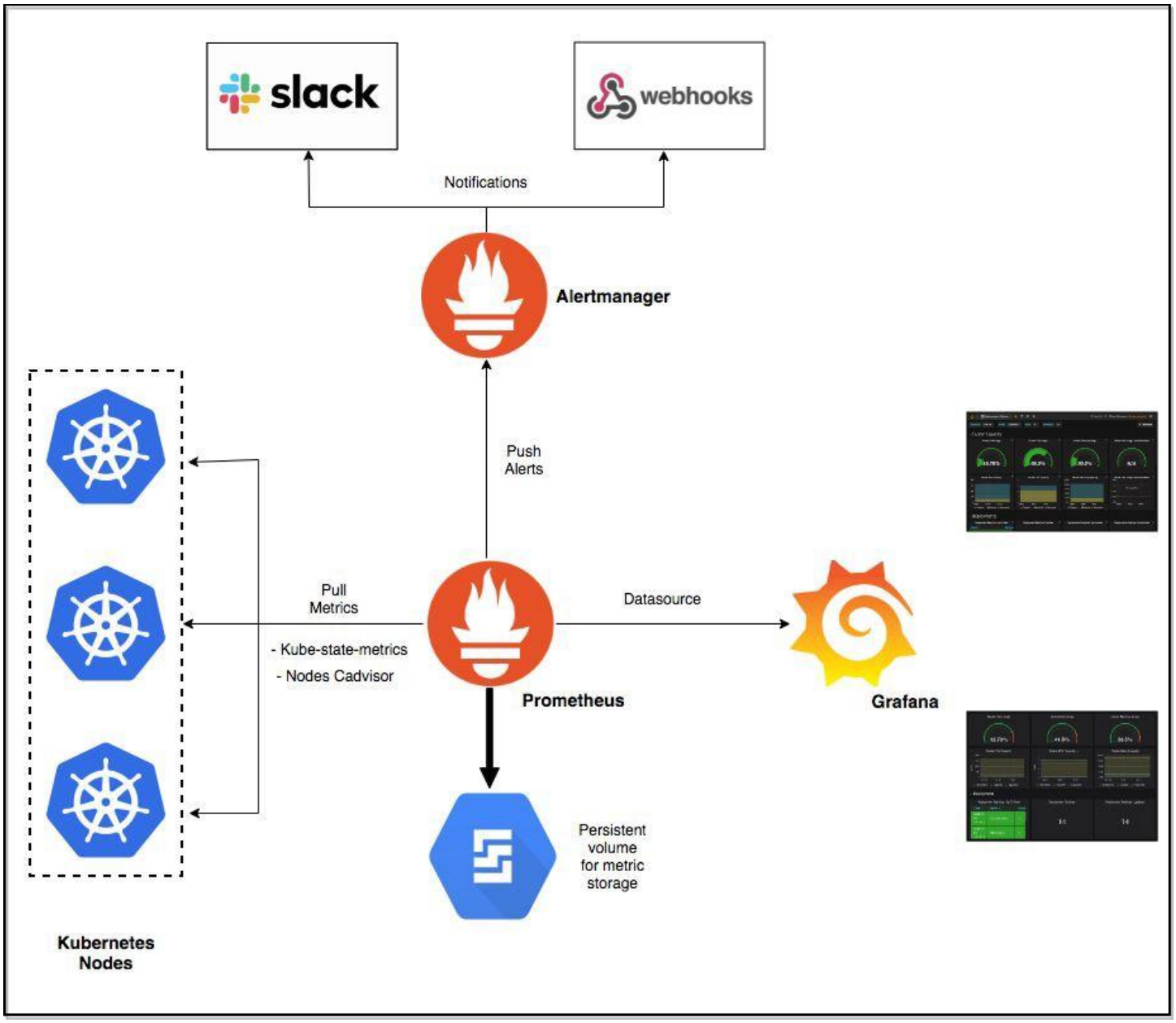

Fig. Prometheus pulling metrics from Kubernetes nodes and sending alerts to Slack and webhooks

As shown in the figure above, the Prometheus will be collecting all the Metrics from Kubernetes nodes and will be sending it to the Grafana dashboard and the alert manager to notify the user over slack or any of their prefered application.
In addition to this Prometheus will also be exporting metrics to the OpenTelemetry collector which will help in the training of Kubebot by storing the collected metrics, traces, logs and events to a common data store.

Grafana

Grafana is multi-platform open-source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources.

Fig. Grafana will be used to visualize alerts and show collected metrics and events

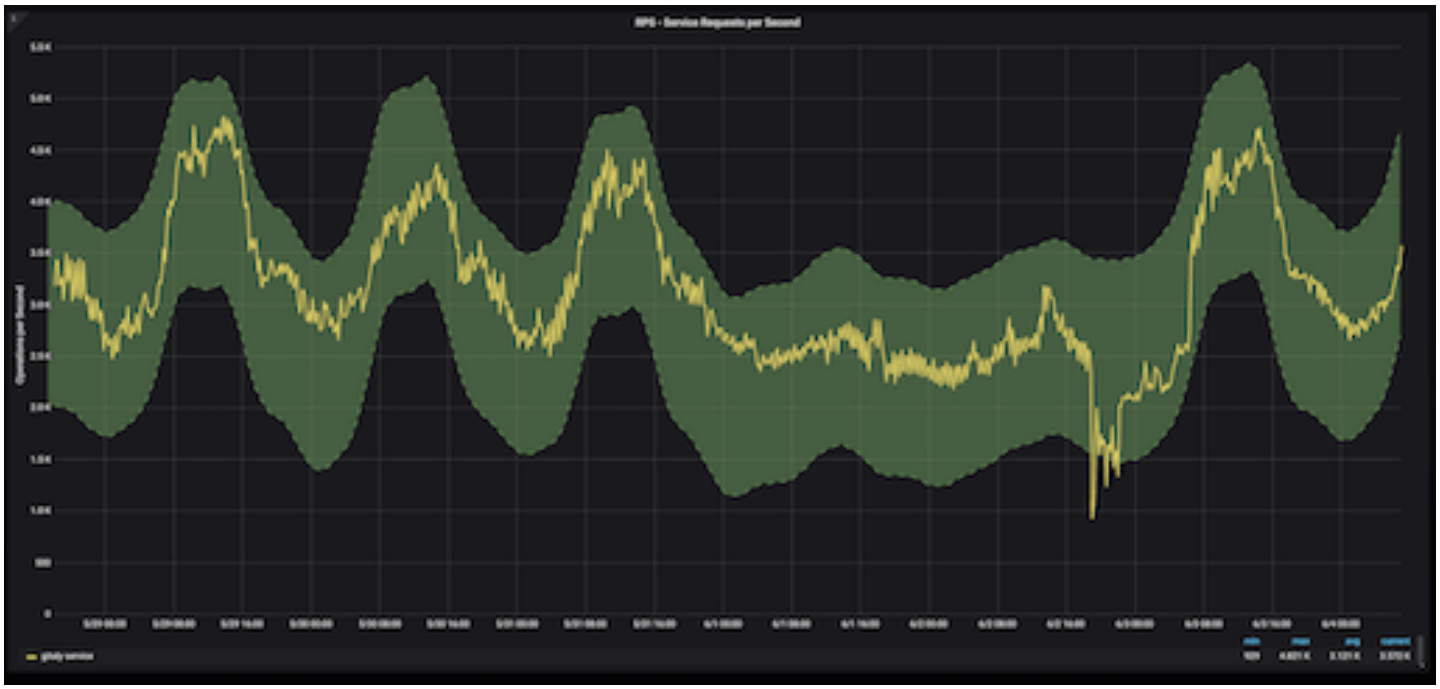

Fig. Grafana graphs will also help the user for easy anomaly detections

Fig. This is what a common Grafana dashboard will look like

As you can see in all the figures the first figure shows the metrics for the user to analyze their applications running in the Kubernetes environment on the go.

The second figure shows anomaly detection, using the similar anomaly detection in the backend the Kubebot will learn using support vector, and reinforcement learning to better detect anomalies.

and in the final figure, the resulting Dashboard will show all the metrics, and the telemetry data along with the relevant alerts on the right side

OpenTelemetry

OpenTelemtry is a collection of tools, APIs, and SDKs. Use it to instrument, generate, collect, and export telemetry data.

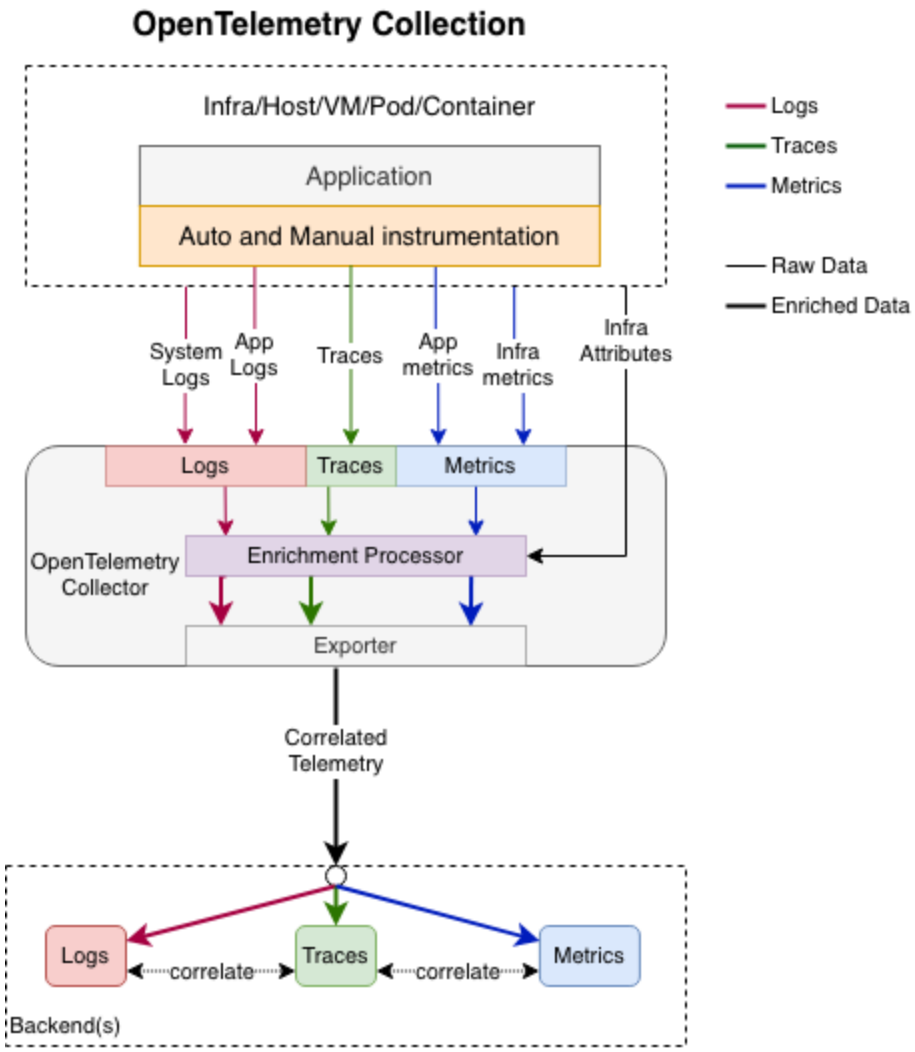

Fig. OpenTelemetry Collection pipeline

The open telemetry will be used to collect filters and do operations on all the collected metrics. It's a common ingestion point for all the collected telemetry data.
The open telemetry collector in kubebot will be responsible for telemetry data filtering and putting all the telemetry data into single storage, the machine learning model will make use of it to train itself and produce better alerts over time.

Fluentbit

Fluent Bit is a super fast, lightweight, and highly scalable logging and metrics processor and forwarder. It is the preferred choice for cloud and containerized environments.

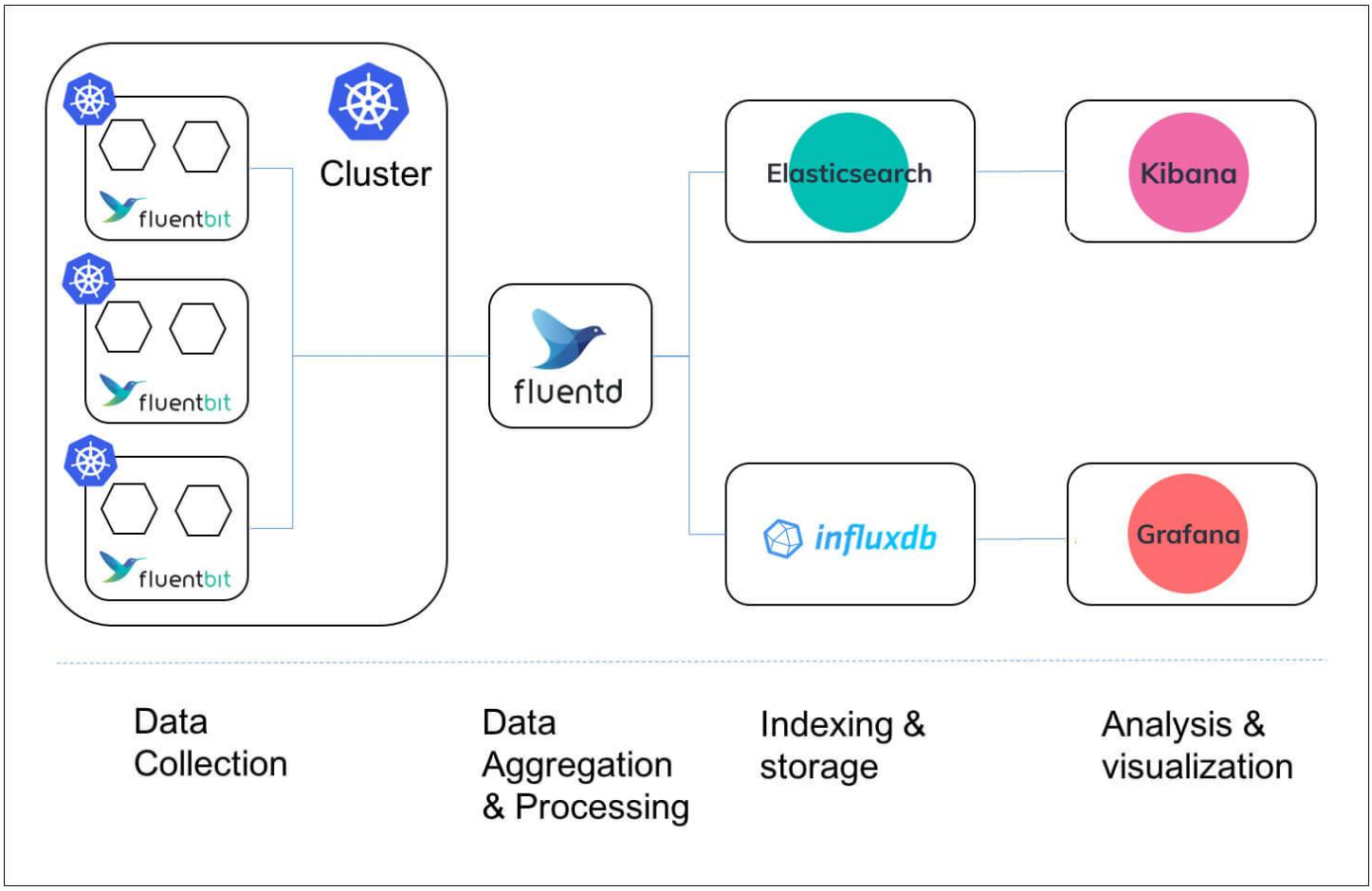

Fig. OpenTelemetry Collection pipeline

In a similar fashion fluent bit will be used in conjunction with the OpenTelemetry collector to collect logs from Kubernetes. The log level will be used by the **machine learning model** to separate fatal errors from info and debug logs, it is one of the most important pillars for KubeBot as logs reveal a lot of important characteristics about what is happening inside Kubernetes environment,

---

Tech Stack:-

1. Golang
2. Kubernetes
3. Prometheus
4. Grafana
5. Fluentbit
6. OpenTelemetry

Timeline:

Note: Each cell contains a time period of 1 week.

S.N.	Date	Work
1)	**20 May - 12 June**	Community Bonding period (take feedback from mentors, finalize architecture)
Coding officially begins!		
2)	**13 June - 26 June**	Setup Kubernetes environment and instrumentation SDKs
3)	**27 June - 10 July**	expose metrics through k8s api server and integrate Prometheus
4)	**11 July - 24 July**	integrate Prometheus with grafana and open telemetry collector
Phase 1 Evaluation (July 25)		
6)	**25 July - 7 Aug**	design alerting and train model
7)	**8 July - 21 Aug**	write end to end unit test
8)	**22 Aug - 4 Sep July**	take feedback and train model
9)	**20 July - 26 July**	integrate slack and other tools for alerting
Final 2 Evaluation (Sept 5)		

Personal Information:

- **Name:** Priyanshu Raj Shrivastava
- **Email:** priyanshuraj400@gmail.com
- **GitHub:** priyanshuraj400
- **LinkedIn:** priyanshuraj400
- **Gitter nickname:** priyanshuraj400
- **First language:** Hindi, proficient in English
- **Time zone:** Indian Standard Time (GMT +5:30)
- **Contact:** +91 8602500319
- **University:** Indian Institute of Technology, Jodhpur

Few sentences about the student and why he/she thinks as the best person to do this project.

My contributions to open source have helped me gain experience in understanding the flow of any pre-written code at a rapid pace and enabled me to add/update features. My previous involvement in understanding Kubernetes includes various code patches among various open-source projects. I have all the prerequisite knowledge about the **Kubernetes**, **docker**, **monitoring** and **machine learning** which is required for the project. I'm also proficient in **Golang** which is required to build such **microservice** architecture and **scalable pipelines**.
Also, I have already completed some projects which require the skills required to complete this project. Further, I don't have any planned commitments during these summers, so I can devote my maximum time to this project.

Questions:

Project URL: https://github.com/leopardslab/kubebot

1). Are you a SCoRe contributor/ Have you contributed to SCoRe before?
Ans: I'm new to the Score community and have found my fit in KubeBot, I have the required skill to take the project forward, I've interacted with mentors and other community members over slack and looking forward to contributing more to Scorelab through GSoC. I am a freelancer and I have been working in cloud services for some time now.

2). How can we reach you (eg: email) if we have questions about your application?
Ans: Email: priyanshuraj400@gmail.com
LinkedIn: priyanshuraj400
Contact: +91 8602500319

3). What is your GitHub username(s):
Ans: GitHub: priyanshuraj400

Project Specific Questions:

4). **Which SCoRe GSoC project are you applying for (please submit separate applications for each project):**
Ans: KubeBot

5) If you have your own project to propose, please describe it here:
Ans: NA

7). List down any plans you have during this summer
Ans: I have no secondary commitments during GSoC period

8). Education:
Ans: Indian Institute Of Technology (IIT), Jodhpur. 2018-2020 | CGPA: 7.63/10
Research Project: Logistics Truck Planning AI Model
• Developing an AI model for logistics truck requirement forecasting, planning and scheduling to facilitate pre-ordering of trucks
• Developed an ML model for fundamental problem statement with 23% cost saving efficacy compared to current truck scheduling costs

10). Do you have work experience in programming? Tell us about it.
Ans: **NOVEL ALGORITHM FOR DIMENSIONAL SYNTHESIS 2019-20 | IIT-J**
Technologies - (Golang, MATLAB, Optimization Techniques, Python)
• Developed a novel algorithm for dimensional synthesis of 4-bar mechanism to trace a desired path. Improved efficacy by implementing Genetic Algorithm, Gradient Descent and fmincon for optimization
• Developed an ML model based on the algorithm to generate the linkage's dimensions. Implemented the technique to develop a Bio-Mechanical Jaw Rehabilitation therapeutic device for TMJ Hypomobility patients

PROGRAMMING ML OPTIMIZATION ALGORITHMS SEP 2020 | IIT-G
Technologies - (Golang, C++, MATLAB)
Built programs on C++ and MATLAB for optimization of Sum Squares Function, Rosenbrock Function, Dixon-Price Function, Trid Function & Zakharov Function using Marquardt's Method and Steepest Descent Method

ML MODEL FOR MINIMIZING 3-D PRINTING STAIRCASE ERROR
Technologies - (MATLAB) | Sep 2019 | IIT-J
Developed a novel algorithm for 3-D printer to optimize printing time with minimum staircase error by using adaptive slicing technique

11). Do you have previous open source experience? Briefly describe what you have done.
Ans:

12) Tell one interesting fact about yourself.
Ans: I am a very curious person and always ready to learn something new. In Lockdown, I have learned playing guitar, practiced meditation and yoga. I have also done a lot of competitive programming.
