IMPLEMENTING OBSERVABILITY IN A SYSTEM USING PROMETHEUS AND GRAFANA

by

MANPREET SINGH

Submitted in partial fulfilment of the requirements for the degree of

MASTER OF ENGINEERING

Major Subject: Internetworking

at

DALHOUSIE UNIVERSITY
Halifax, Nova Scotia
July, 2024

Dalhousie University

Faculty of Engineering

Internetworking

The undersigned hereby certify that they have read and awarded a pass in INWK 6314 for the course/seminar entitled "IMPLEMENTING OBSERVABILITY IN A SYSTEM USING PROMETHEUS AND GRAFANA" by *Manpreet Singh* in partial fulfilment of the requirements for the degree of Master of Engineering.

_____

DALHOUSIE UNIVERSITY
INTERNETWORKING PROGRAM

AUTHORITY TO DISTRIBUTE REPORT

Title:

IMPLEMENTING OBSERVABILITY IN A SYSTEM USING PROMETHEUS AND GRAFANA

The Internetworking Program may make available or authorise others to make available individual photo/microfilm or soft copies of this report without restrictions after insert date here.

The author attests that permission has been obtained for the use of any copyrighted material appearing in this report (other than brief excerpts requiring only proper acknowledgement in scholarly writing) and that all such use is clearly acknowledged.

Full Name of Author:

Signature of Author: Manpreet Singh
Date: July 5, 2024

Table of Contents

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Dr. Hasan Raza for his invaluable guidance and support throughout the preparation of my project report. Your expertise and insightful feedback have been instrumental in shaping the document into its final form.

Your willingness to share your knowledge and provide constructive criticism have not only enhanced my understanding of the subject matter but have also significantly improved the quality of my work. Your mentorship has been a source of inspiration, and I am truly grateful for the time and effort you have invested in helping me succeed.

Thank you once again for your dedication and commitment to your students' academic growth. I look forward to applying the skills and knowledge I've gained under your guidance in my future endeavors.

EXECUTIVE SUMMARY

Implementing observability into a system involves creating the capability to measure, monitor, and understand the internal state of the system. The implementation involves defining key metrics, storing the relevant information in a time series database, setting up dashboards for clarity in visualization(using Grafana) and Alert configurations using Prometheus Alertmanager based on defined Threshold/Metric parameters.

Metrics are numerical values that can be measured and tracked over time. They provide objective data points that reflect the state of the system.

Observability holds a significant space in the field of platform engineering as without observability, it is challenging to diagnose and troubleshoot issues quickly. When a problem occurs, it can be difficult to pinpoint the root cause, leading to increased downtime and frustration. Lack of metrics, logs, and traces means limited visibility into the system's behavior and performance, making it hard to understand what went wrong

The objective of the project is to delve into the workings of Prometheus and Grafana tools by scraping the metrics of the localhost where the system is deployed and generating alerts thereby notifying a threshold condition is reached and an appropriate action is required to bring the system back into optimal state.

1 INTRODUCTION

Prometheus is a robust open-source system and time-series database specifically designed for monitoring and observability. Developed initially by SoundCloud, Prometheus has emerged as a cornerstone in the landscape of cloud-native computing and DevOps practices. It excels in collecting metrics from diverse sources, empowering teams to monitor everything from infrastructure performance to application behavior in real-time. With its multidimensional data model, powerful query language (PromQL), and efficient storage, Prometheus enables proactive monitoring and troubleshooting across dynamic, distributed environments. Its ecosystem thrives on integrations with a wide array of software systems, fostering a community-driven approach that continuously evolves to meet the growing demands of scalability, reliability, and actionable insights in today's complex technological landscapes. Thus, Prometheus stands not only as a tool but as a symbol of empowerment for modern engineering teams striving for comprehensive observability and operational excellence

1.1 The Need For Observability

Observability in a system refers to the ability to understand and diagnose what is happening inside the system based on its external outputs or behaviors. It is a critical aspect of system design and operation, particularly in complex and distributed systems. Here are several key reasons why observability is essential:

- Diagnosing Issues: In any system, problems can arise unexpectedly. Observability allows engineers and operators to monitor the system's behavior in real-time or retrospectively. This monitoring helps identify the root cause of issues quickly, leading to faster resolutions and minimizing downtime.
- Understanding Performance: Observability provides insights into how the system is performing under normal conditions and during peak loads or stress. This visibility helps in optimizing performance and resource allocation, ensuring efficient use of system resources.
- Monitoring User Experience: For systems that interact directly with users (such as web applications or customer service platforms), observability helps track and understand user interactions and experiences. This information is crucial for improving user satisfaction and addressing usability issues.

- Analyzing Trends and Patterns: By collecting and analyzing observational data over time, patterns and trends can be identified. These insights enable proactive decision-making, such as capacity planning, feature enhancements, or detecting potential security threats.
- Debugging and Iterative Development: During the development phase, observability tools aid in debugging code and verifying new features. Developers can monitor how changes affect the system and validate performance against expected outcomes.

1.2 Implementing Observability On Localhost

In order to execute the project, we have deployed a stack of services on the local host. The official documentation(https://prometheus.io/) was referenced extensively to understand the depth of the system.

2 Components Of Prometheus

The emphasis of this section is to understand the core components of the Prometheus tool. We have divided this section into subsections to describe the different components Prometheus provides to implement observability.

2.1 Prometheus Server

Prometheus Server stands as the cornerstone of the Prometheus observability ecosystem, pivotal in modern cloud-native monitoring and alerting practices. Functioning as both collector and storage engine, it efficiently gathers time-series metrics from diverse targets via HTTP endpoints, employing a multi-dimensional data model with labels for flexible querying and aggregation. Supported by PromQL, its powerful query language, Prometheus Server enables detailed analysis and real-time monitoring of application and infrastructure performance. With robust alerting rules and integration with Alertmanager for streamlined alert handling, it empowers proactive incident response. Its scalable architecture, coupled with support for various service discovery mechanisms and exporters, ensures adaptability to dynamic environments.

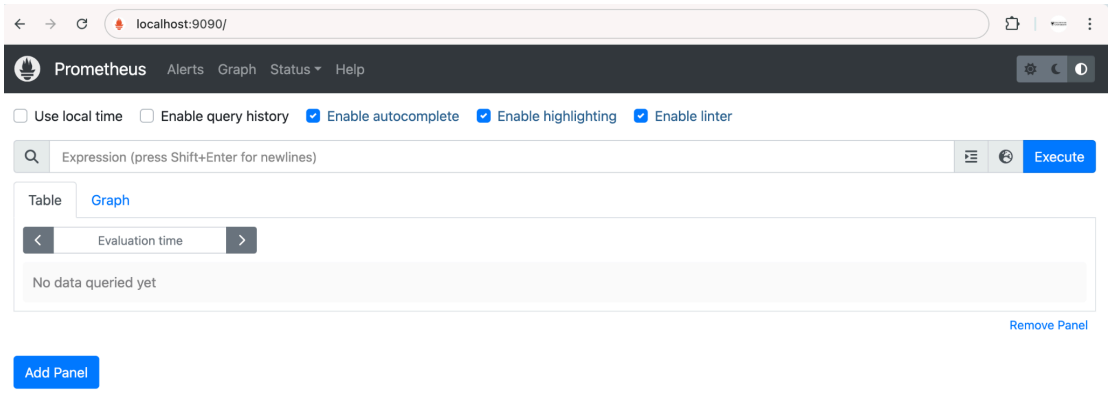
Prometheus Server provides a built-in web-based user interface (UI) for exploring metrics, running queries, and visualizing data. Additionally, it exposes a comprehensive HTTP API that allows programmatic access for querying metrics, managing alerting rules, and more.

The Prometheus server runs on port 9090.

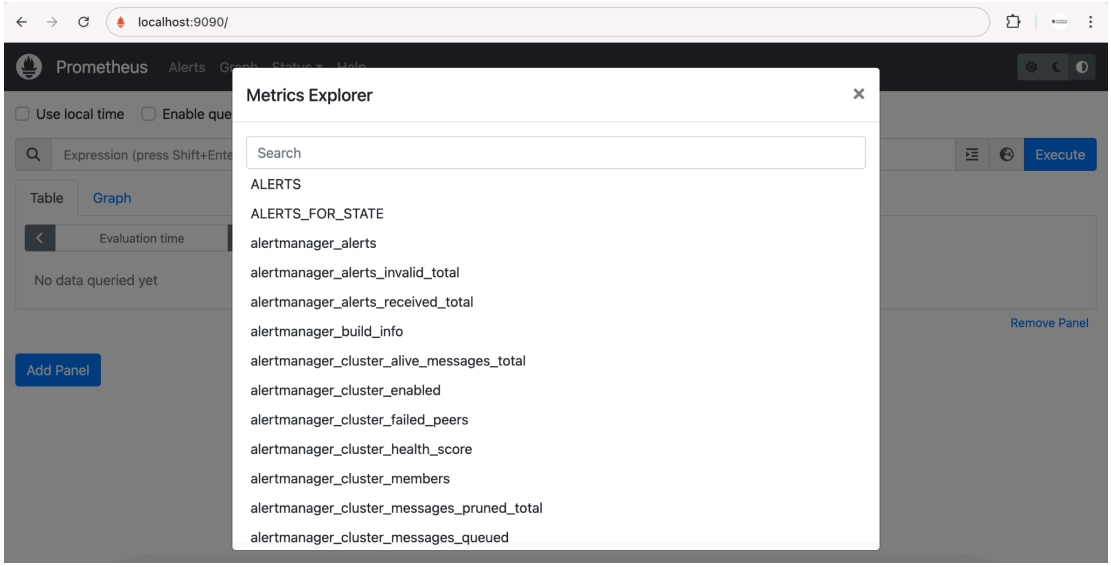
Prometheus service running on localhost:

```
manpreetsingh@tc69f prometheus-2.53.0.darwin-amd64 % ./prometheus
ts=2024-07-05T18:40:43.606Z caller=main.go:589 level=info msg="No time or size retention was set so using the default time retention" duration=15d
ts=2024-07-05T18:40:43.606Z caller=main.go:633 level=info msg="Starting Prometheus Server" mode=server version="(version=2.53.0, branch=HEAD, revision=4c35b9250afefede41c5f5acd76191
f90f625898)"
ts=2024-07-05T18:40:43.606Z caller=main.go:638 level=info build_context="(go=go1.22.4, platform=darwin/amd64, user=root@be0036ec0ddd, date=20240619-07:39:13, tags=netgo,builtinasset
s,stringlabels)"
ts=2024-07-05T18:40:43.606Z caller=main.go:639 level=info host_details=(darwin)
ts=2024-07-05T18:40:43.606Z caller=main.go:640 level=info fd_limits="(soft=10240, hard=unlimited)"
ts=2024-07-05T18:40:43.606Z caller=main.go:641 level=info vm_limits="(soft=unlimited, hard=unlimited)"
ts=2024-07-05T18:40:43.611Z caller=web.go:568 level=info component=web msg="Start listening for connections" address=0.0.0.0:9090
ts=2024-07-05T18:40:43.612Z caller=main.go:1148 level=info msg="Starting TSDB ..."
ts=2024-07-05T18:40:43.612Z caller=repair.go:56 level=info component=tsdb msg="Found healthy block" mint=1719941722854 maxt=1719986400000 ulid=01J1W76VFYQ5405S6W1NKARVBJ
ts=2024-07-05T18:40:43.612Z caller=repair.go:56 level=info component=tsdb msg="Found healthy block" mint=1719986878865 maxt=1720051200000 ulid=01J1YZVM0RGVY94BQMD381WJZZ
ts=2024-07-05T18:40:43.612Z caller=repair.go:56 level=info component=tsdb msg="Found healthy block" mint=1720051205700 maxt=1720101600000 ulid=01J205ZGAW6ZF2551H7G0TNXPN
ts=2024-07-05T18:40:43.612Z caller=repair.go:56 level=info component=tsdb msg="Found healthy block" mint=1720118471703 maxt=1720130400000 ulid=01J20QP6NY4F2SRNS4STYZEDGQ
ts=2024-07-05T18:40:43.613Z caller=repair.go:56 level=info component=tsdb msg="Found healthy block" mint=1720159244837 maxt=1720166400000 ulid=01J21J80PHHMWEN39TSVPPCE6P
ts=2024-07-05T18:40:43.613Z caller=repair.go:56 level=info component=tsdb msg="Found healthy block" mint=1720183509001 maxt=1720188000000 ulid=01J222RX0A5MJ7AGRKPQ2YRVKW
ts=2024-07-05T18:40:43.613Z caller=repair.go:56 level=info component=tsdb msg="Found healthy block" mint=1720140668785 maxt=1720159200000 ulid=01J222RZNER7MZ4C8Z1382595N
ts=2024-07-05T18:40:43.614Z caller=tls_config.go:313 level=info component=web msg="Listening on" address=[::]:9090
ts=2024-07-05T18:40:43.614Z caller=tls_config.go:316 level=info component=web msg="TLS is disabled." http2=false address=[::]:9090
ts=2024-07-05T18:40:43.622Z caller=head.go:626 level=info component=tsdb msg="Replaying on-disk memory mappable chunks if any"
ts=2024-07-05T18:40:43.622Z caller=head.go:713 level=info component=tsdb msg="On-disk memory mappable chunks replay completed" duration=221.876µs
ts=2024-07-05T18:40:43.622Z caller=head.go:721 level=info component=tsdb msg="Replaying WAL, this may take a while"
ts=2024-07-05T18:40:43.630Z caller=head.go:758 level=info component=tsdb msg="WAL checkpoint loaded"
ts=2024-07-05T18:40:43.631Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=45 maxSegment=50
ts=2024-07-05T18:40:43.632Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=46 maxSegment=50
ts=2024-07-05T18:40:43.633Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=47 maxSegment=50
ts=2024-07-05T18:40:43.650Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=48 maxSegment=50
ts=2024-07-05T18:40:43.651Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=49 maxSegment=50
ts=2024-07-05T18:40:43.651Z caller=head.go:793 level=info component=tsdb msg="WAL segment loaded" segment=50 maxSegment=50
ts=2024-07-05T18:40:43.651Z caller=head.go:830 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=7.870388ms wal_replay_duration=21.078853ms wbl_replay_
duration=170ns chunk_snapshot_load_duration=0s mmap_chunk_replay_duration=221.876µs total_replay_duration=29.221683ms
ts=2024-07-05T18:40:43.655Z caller=main.go:1169 level=info fs_type=1a
ts=2024-07-05T18:40:43.655Z caller=main.go:1172 level=info msg="TSDB started"
ts=2024-07-05T18:40:43.656Z caller=main.go:1354 level=info msg="Loading configuration file" filename=prometheus.yml
ts=2024-07-05T18:40:43.691Z caller=main.go:1391 level=info msg="updated GOGC" old=100 new=75
ts=2024-07-05T18:40:43.691Z caller=main.go:1402 level=info msg="Completed loading of configuration file" filename=prometheus.yml totalDuration=35.2153ms db_storage=989ns remote_stor
age=3.918µs web_handler=436ns query_engine=2.892µs scrape=33.400425ms scrape_sd=137.662µs notify=67.4µs notify_sd=20.532µs rules=1.130608ms tracing=19.956µs
ts=2024-07-05T18:40:43.691Z caller=main.go:1133 level=info msg="Server is ready to receive web requests."
ts=2024-07-05T18:40:43.691Z caller=manager.go:164 level=info component="rule manager" msg="Starting rule manager..."
ts=2024-07-05T18:40:49.315Z caller=compact.go:567 level=info component=tsdb msg="write block resulted in empty block" mint=1720188000000 maxt=1720195200000 duration=75.379735ms
ts=2024-07-05T18:40:49.319Z caller=head.go:1355 level=info component=tsdb msg="Head GC completed" caller=truncateMemory duration=3.122621ms
ts=2024-07-05T18:40:49.320Z caller=checkpoint.go:101 level=info component=tsdb msg="Creating checkpoint" from_segment=45 to_segment=47 mint=1720195200000
ts=2024-07-05T18:40:49.373Z caller=head.go:1317 level=info component=tsdb msg="WAL checkpoint complete" first=45 last=47 duration=53.722165ms
```

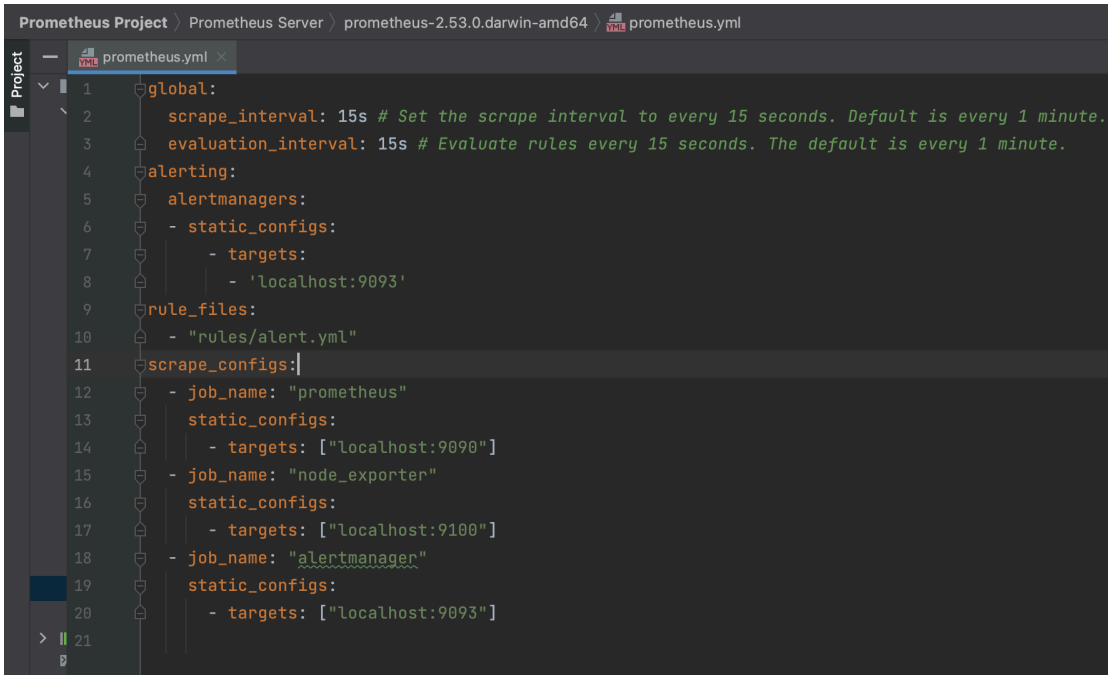
Prometheus UI:



Metrics Exposed From Prometheus:



Prometheus Server Configuration File:



2.2 PromQL

PromQL or Prometheus Query Language is a powerful query language designed for working with Prometheus data. It allows users to write expressive queries to retrieve, manipulate, and aggregate time-series data based on specific criteria

It is important to reveal all the details of the problem to the reader. This chapter is used to make the reader aware of all the nuances of the problem so that any subsequent trade-offs or measurements are justifiable in the context of the problem desc

2.3 Time-Series Database

Prometheus Server maintains a local, efficient time-series database optimized for storing metrics collected from various targets. This database structure allows for high write throughput and efficient querying of historical data

2.4 Exporters

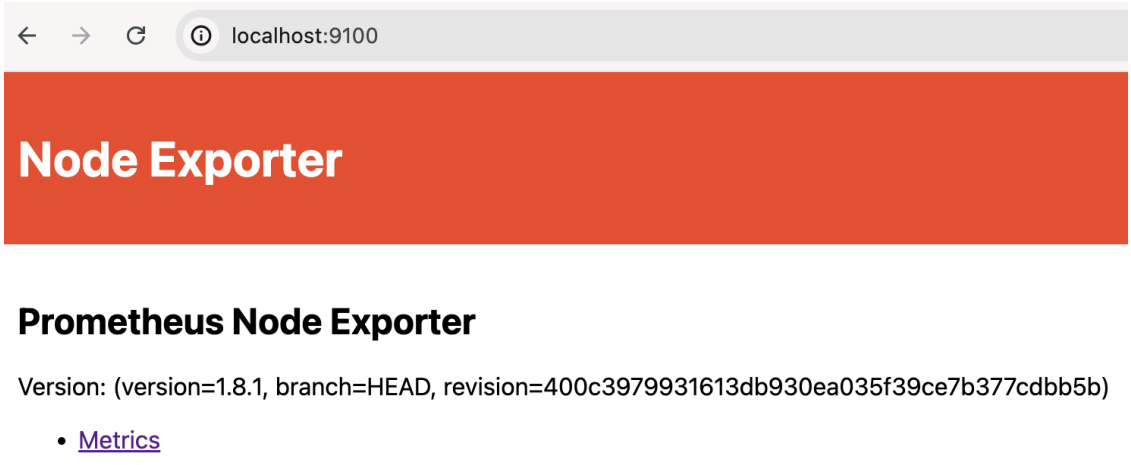
Exporters in Prometheus play a pivotal role in extending the monitoring capabilities of the Prometheus ecosystem by enabling the collection of metrics from a diverse array of systems and services that do not natively expose Prometheus-compatible metrics. These exporters act as intermediaries, translating metrics from various formats or protocols into a format Prometheus can scrape and store in its time-series database. They facilitate seamless integration with a wide range of systems, including databases (such as MySQL, PostgreSQL), web servers (like Apache and Nginx), message brokers (such as RabbitMQ and Kafka), cloud platforms (like AWS and Google Cloud), and numerous other applications and services crucial to modern IT infrastructures. Each exporter is specifically designed to understand the metrics format of the system it monitors and exposes these metrics via an HTTP endpoint typically located at `/metrics`. Prometheus scrapes these endpoints at configured intervals, allowing it to collect metrics on resource utilization, performance indicators, operational metrics, and more. Popular examples of exporters include the Node Exporter for system metrics like CPU, memory, and disk usage; the Blackbox Exporter for probing endpoints via HTTP, ICMP, TCP, and DNS; and the JMX

We have considered the use case of deploying NodeExporter on local host which runs the service on port 9100 and also exposes a UI on http://localhost:9100

NodeExporter Service Deployed On Localhost:

```
manpreetsingh@tc69f node_exporter-1.8.1.darwin-amd64 % ./node_exporter
ts=2024-07-03T23:19:37.751Z caller=node_exporter.go:193 level=info msg="Starting node_exporter" version="(version=1.8.1, branch=HEAD, revision=400c3
979931613db930ea035f39ce7b377cdbb5b)"
ts=2024-07-03T23:19:37.755Z caller=node_exporter.go:194 level=info msg="Build context" build_context="(go=go1.22.3, platform=darwin/amd64, user=root
@eea0d40c3e38, date=20240521-18:39:19, tags=unknown)"
ts=2024-07-03T23:19:37.756Z caller=filesystem_common.go:111 level=info collector=filesystem msg="Parsed flag --collector.filesystem.mount-points-exc
lude" flag=^/(dev)($|/)
ts=2024-07-03T23:19:37.757Z caller=filesystem_common.go:113 level=info collector=filesystem msg="Parsed flag --collector.filesystem.fs-types-exclude
" flag=^devfs$
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:111 level=info msg="Enabled collectors"
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=boottime
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=cpu
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=diskstats
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=filesystem
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=loadavg
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=meminfo
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=netdev
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=os
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=powersupplyclass
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=textfile
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=thermal
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=time
ts=2024-07-03T23:19:37.757Z caller=node_exporter.go:118 level=info collector=uname
ts=2024-07-03T23:19:37.776Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9100
ts=2024-07-03T23:19:37.776Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:9100
```

NodeExporter UI Exposed On Port 9100:



'Metrics' would display a list of all the metrics which are exposed from the NodeExporter thereby exposing time-series data points of various metrics of the localhost.

2.5 Alertmanager

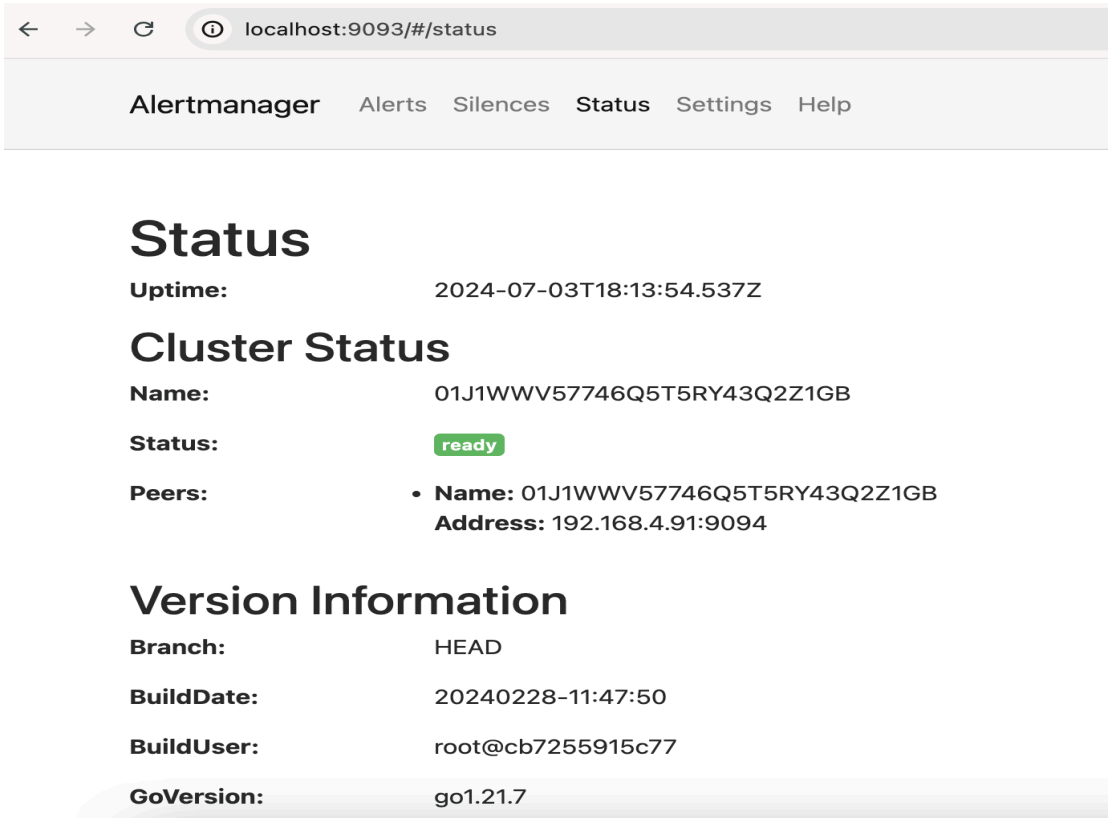
Prometheus Server supports alerting based on predefined rules and thresholds. When conditions specified in alerting rules are met, Prometheus triggers alerts. These alerts are then managed and routed by Alertmanager, a separate component closely integrated with Prometheus Server. This component runs on port 9093 and has its own configuration file to configure various alerts and send the alerts to an external system.

Keeping the scope of the project in mind, I have configured gmail as the external system receiving the alerts when localhost's CPU utilization crosses 20% utilization.

AlertManager Running On Port 9093 on Localhost:

```
manpreetsingh@Manpreets-MBP alertmanager-0.27.0.darwin-amd64 %
manpreetsingh@Manpreets-MBP alertmanager-0.27.0.darwin-amd64 % ./alertmanager
ts=2024-07-03T02:33:22.896Z caller=main.go:181 level=info msg="Starting Alertmanager" version="(version=0.27.0, branch=HEAD, revision=0aa3c2aad14cff039931923ab16b26b7481783b5)"
ts=2024-07-03T02:33:22.897Z caller=main.go:182 level=info build_context="(go=go1.21.7, platform=darwin/amd64, user=root@cb7255915c77, date=20240228-11:47:50, tags=netgo)"
ts=2024-07-03T02:33:22.913Z caller=cluster.go:186 level=info component=cluster msg="setting advertise address explicitly" addr=192.168.4.91 port=9094
ts=2024-07-03T02:33:22.920Z caller=cluster.go:683 level=info component=cluster msg="Waiting for gossip to settle..." interval=2s
ts=2024-07-03T02:33:22.971Z caller=coordinator.go:113 level=info component=configuration msg="Loading configuration file" file=alertmanager.yml
ts=2024-07-03T02:33:22.974Z caller=coordinator.go:126 level=info component=configuration msg="Completed loading of configuration file" file=alertmanager.yml
ts=2024-07-03T02:33:22.980Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9093
ts=2024-07-03T02:33:22.980Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:9093
ts=2024-07-03T02:33:24.925Z caller=cluster.go:708 level=info component=cluster msg="gossip not settled" polls=0 before=0 now=1 elapsed=2.004546243s
ts=2024-07-03T02:33:32.927Z caller=cluster.go:700 level=info component=cluster msg="gossip settled; proceeding" elapsed=10.007156277s
^Cts=2024-07-03T16:41:41.255Z caller=main.go:542 level=info msg="Received SIGTERM, exiting gracefully..."
manpreetsingh@Manpreets-MBP alertmanager-0.27.0.darwin-amd64 % ./alertmanager
ts=2024-07-03T16:41:42.329Z caller=main.go:181 level=info msg="Starting Alertmanager" version="(version=0.27.0, branch=HEAD, revision=0aa3c2aad14cff039931923ab16b26b7481783b5)"
ts=2024-07-03T16:41:42.330Z caller=main.go:182 level=info build_context="(go=go1.21.7, platform=darwin/amd64, user=root@cb7255915c77, date=20240228-11:47:50, tags=netgo)"
ts=2024-07-03T16:41:42.339Z caller=cluster.go:186 level=info component=cluster msg="setting advertise address explicitly" addr=192.168.4.91 port=9094
ts=2024-07-03T16:41:42.340Z caller=cluster.go:683 level=info component=cluster msg="Waiting for gossip to settle..." interval=2s
ts=2024-07-03T16:41:42.375Z caller=coordinator.go:113 level=info component=configuration msg="Loading configuration file" file=alertmanager.yml
ts=2024-07-03T16:41:42.375Z caller=coordinator.go:126 level=info component=configuration msg="Completed loading of configuration file" file=alertmanager.yml
ts=2024-07-03T16:41:42.381Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9093
ts=2024-07-03T16:41:42.381Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:9093
ts=2024-07-03T16:41:44.341Z caller=cluster.go:708 level=info component=cluster msg="gossip not settled" polls=0 before=0 now=1 elapsed=2.000754005s
ts=2024-07-03T16:41:52.348Z caller=cluster.go:700 level=info component=cluster msg="gossip settled; proceeding" elapsed=10.007076664s
^Cts=2024-07-03T16:51:13.695Z caller=main.go:542 level=info msg="Received SIGTERM, exiting gracefully..."
manpreetsingh@Manpreets-MBP alertmanager-0.27.0.darwin-amd64 % ./alertmanager
ts=2024-07-03T16:51:15.876Z caller=main.go:181 level=info msg="Starting Alertmanager" version="(version=0.27.0, branch=HEAD, revision=0aa3c2aad14cff039931923ab16b26b7481783b5)"
ts=2024-07-03T16:51:15.876Z caller=main.go:182 level=info build_context="(go=go1.21.7, platform=darwin/amd64, user=root@cb7255915c77, date=20240228-11:47:50, tags=netgo)"
ts=2024-07-03T16:51:15.886Z caller=cluster.go:186 level=info component=cluster msg="setting advertise address explicitly" addr=192.168.4.91 port=9094
ts=2024-07-03T16:51:15.888Z caller=cluster.go:683 level=info component=cluster msg="Waiting for gossip to settle..." interval=2s
ts=2024-07-03T16:51:15.928Z caller=coordinator.go:113 level=info component=configuration msg="Loading configuration file" file=alertmanager.yml
ts=2024-07-03T16:51:15.929Z caller=coordinator.go:126 level=info component=configuration msg="Completed loading of configuration file" file=alertmanager.yml
ts=2024-07-03T16:51:15.934Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9093
ts=2024-07-03T16:51:15.934Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:9093
ts=2024-07-03T16:51:17.891Z caller=cluster.go:708 level=info component=cluster msg="gossip not settled" polls=0 before=0 now=1 elapsed=2.002616762s
ts=2024-07-03T16:51:25.894Z caller=cluster.go:700 level=info component=cluster msg="gossip settled; proceeding" elapsed=10.0059738332s
^Cts=2024-07-03T16:53:55.501Z caller=main.go:542 level=info msg="Received SIGTERM, exiting gracefully..."
manpreetsingh@Manpreets-MBP alertmanager-0.27.0.darwin-amd64 % ./alertmanager
ts=2024-07-03T16:53:56.485Z caller=main.go:181 level=info msg="Starting Alertmanager" version="(version=0.27.0, branch=HEAD, revision=0aa3c2aad14cff039931923ab16b26b7481783b5)"
ts=2024-07-03T16:53:56.485Z caller=main.go:182 level=info build_context="(go=go1.21.7, platform=darwin/amd64, user=root@cb7255915c77, date=20240228-11:47:50, tags=netgo)"
ts=2024-07-03T16:53:56.499Z caller=cluster.go:186 level=info component=cluster msg="setting advertise address explicitly" addr=192.168.4.91 port=9094
ts=2024-07-03T16:53:56.501Z caller=cluster.go:683 level=info component=cluster msg="Waiting for gossip to settle..." interval=2s
ts=2024-07-03T16:53:56.536Z caller=coordinator.go:113 level=info component=configuration msg="Loading configuration file" file=alertmanager.yml
ts=2024-07-03T16:53:56.537Z caller=coordinator.go:126 level=info component=configuration msg="Completed loading of configuration file" file=alertmanager.yml
ts=2024-07-03T16:53:56.542Z caller=tls_config.go:313 level=info msg="Listening on" address=[::]:9093
ts=2024-07-03T16:53:56.542Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false address=[::]:9093
ts=2024-07-03T16:53:58.504Z caller=cluster.go:708 level=info component=cluster msg="gossip not settled" polls=0 before=0 now=1 elapsed=2.002752747s
ts=2024-07-03T16:54:06.508Z caller=cluster.go:700 level=info component=cluster msg="gossip settled; proceeding" elapsed=10.007317244s
^Cts=2024-07-03T17:00:09.753Z caller=main.go:542 level=info msg="Received SIGTERM, exiting gracefully..."
manpreetsingh@Manpreets-MBP alertmanager-0.27.0.darwin-amd64 % ./alertmanager
ts=2024-07-03T17:00:11.478Z caller=main.go:181 level=info msg="Starting Alertmanager" version="(version=0.27.0, branch=HEAD, revision=0aa3c2aad14cff039931923ab16b26b7481783b5)"
ts=2024-07-03T17:00:11.478Z caller=main.go:182 level=info build_context="(go=go1.21.7, platform=darwin/amd64, user=root@cb7255915c77, date=20240228-11:47:50, tags=netgo)"
ts=2024-07-03T17:00:11.493Z caller=cluster.go:186 level=info component=cluster msg="setting advertise address explicitly" addr=192.168.4.91 port=9094
```

AlertManagerUI:



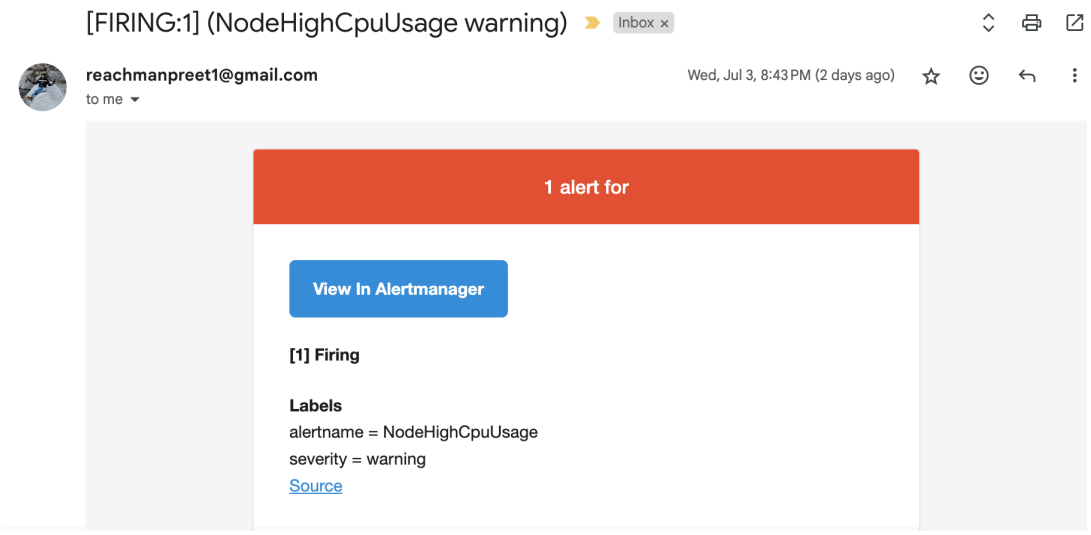
Alerts Configured Referring To NodeExporter Metrics:

prometheus.yml | alert.yml

```yaml
groups:
- name: my-rules
  rules:
  - alert: NodeExporterDown
    expr: up{job="node_exporter"} == 0

  - alert: NodeHighCpuUsage
    expr: 100 * (1 - avg(irate(node_cpu_seconds_total{mode="idle"}[1m]))) > 20
    for: 1m
    labels:
      severity: warning

  - alert: NodeHighMemoryUsage
    expr: node_memory_MemUsed_percent > 80
    for: 5m
    labels:
      severity: critical

```

This alert.yml file is then pointed into the Prometheus.yaml file so prometheus can read the alerts and fire them to AlertManager

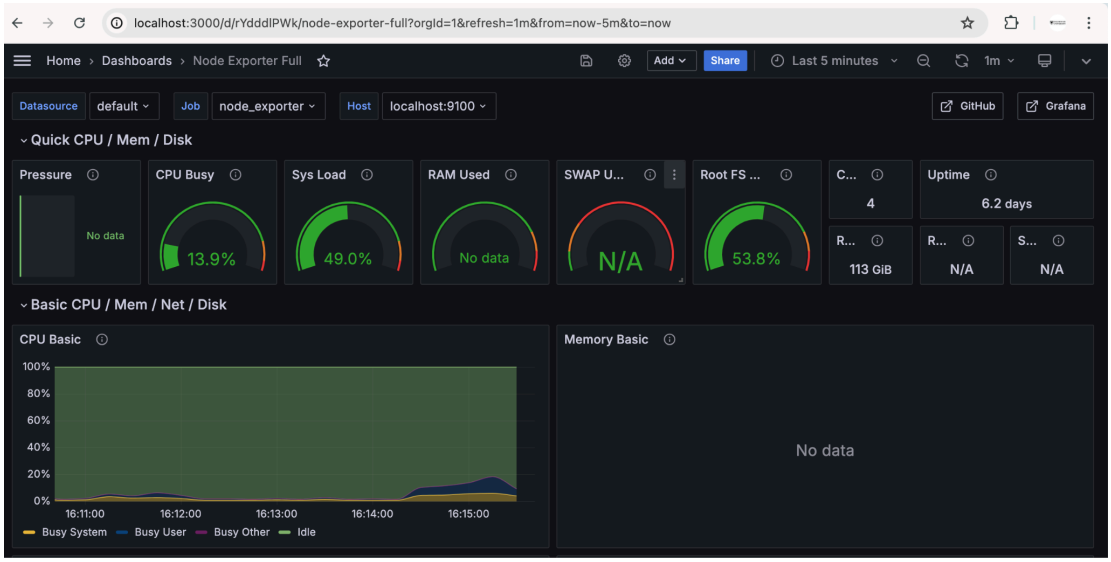
Alerts Triggered To The Gmail System From AlertManager:



2.6 Integrating Grafana For Data Visualization

We have imported the NodeExporter and Prometheus templates in our project in the Grafana tool which runs on port 3000 on the localhost.

Grafana Tool Exposing The NodeExporter Metrics:



Grafana Service Running On Port 3000 On LocalHost:

```
manpreetsingh@tc69f grafana-v11.1.0 % ./bin/grafana-server web

Deprecation warning: The standalone 'grafana-server' program is deprecated and will be removed in the future. Please update all uses of 'grafana-server' to 'grafana server'
INFO [07-05|16:16:44] Starting Grafana                         logger=settings version=11.1.0 commit=5b85c4c2fcf5d32d4f68aaef345c53096359b2f1 branch=HEAD compiled=2024-07-05T16:16:4
4+03:00
INFO [07-05|16:16:44] Config loaded from                       logger=settings file="/Users/manpreetsingh/Desktop/Prometheus Project/Prometheus Server/grafana-v11.1.0/conf/defaults.
ini"
INFO [07-05|16:16:44] Target                                   logger=settings target=[all]
INFO [07-05|16:16:44] Path Home                                logger=settings path="/Users/manpreetsingh/Desktop/Prometheus Project/Prometheus Server/grafana-v11.1.0"
INFO [07-05|16:16:44] Path Data                                logger=settings path="/Users/manpreetsingh/Desktop/Prometheus Project/Prometheus Server/grafana-v11.1.0/data"
INFO [07-05|16:16:44] Path Logs                                logger=settings path="/Users/manpreetsingh/Desktop/Prometheus Project/Prometheus Server/grafana-v11.1.0/data/log"
INFO [07-05|16:16:44] Path Plugins                             logger=settings path="/Users/manpreetsingh/Desktop/Prometheus Project/Prometheus Server/grafana-v11.1.0/data/plugins"
INFO [07-05|16:16:44] Path Provisioning                        logger=settings path="/Users/manpreetsingh/Desktop/Prometheus Project/Prometheus Server/grafana-v11.1.0/conf/provision
ing"
INFO [07-05|16:16:44] App mode production                      logger=settings
INFO [07-05|16:16:44] FeatureToggles                           logger=featuremgmt prometheusDataplane=true awsAsyncQueryCaching=true prometheusMetricEncyclopedia=true transformation
sRedesign=true panelMonitoring=true lokiMetricDataplane=true alertingNoDataErrorExecution=true managedPluginsInstall=true logRowsPopoverMenu=true angularDeprecationUI=true ssoSettin
gsApi=true kubernetesPlaylists=true alertingInsights=true cloudWatchNewLabelParsing=true recoveryThreshold=true lokiQueryHints=true dataplaneFrontendFallback=true lokiStructuredMeta
data=true logsInfiniteScrolling=true dashgpt=true annotationPermissionUpdate=true alertingSimplifiedRouting=true betterPageScrolling=true correlations=true recordedQueriesMulti=true
 nestedFolders=true awsDatasourcesNewFormStyling=true logsExploreTableVisualisation=true prometheusConfigOverhaulAuth=true topnav=true exploreMetrics=true cloudWatchCrossAccountQuer
ying=true influxdbBackendMigration=true logsContextDatasourceUi=true lokiQuerySplitting=true exploreContentOutline=true publicDashboards=true
INFO [07-05|16:16:44] Connecting to DB                         logger=sqlstore dbtype=sqlite3
WARN [07-05|16:16:44] SQLite database file has broader permissions than it should logger=sqlstore path="/Users/manpreetsingh/Desktop/Prometheus Project/Prometheus Server/grafana-v11
.1.0/data/grafana.db" mode=-rwxrwxrwx expected=-rw-r-----
INFO [07-05|16:16:44] Locking database                         logger=migrator
INFO [07-05|16:16:44] Starting DB migrations                   logger=migrator
INFO [07-05|16:16:44] migrations completed                     logger=migrator performed=0 skipped=667 duration=2.249366ms
INFO [07-05|16:16:44] Unlocking database                       logger=migrator
INFO [07-05|16:16:44] Validated license token                  logger=licensing appURL=http://localhost:3000/ source=disk status=NotFound
INFO [07-05|16:16:44] Envelope encryption state                logger=secrets enabled=true current provider=secretKey.v1
ERROR[07-05|16:16:44] Failed to get renderer plugin sources    logger=renderer.manager error="failed to open plugins path"
INFO [07-05|16:16:44] Restored cache from database             logger=plugin.angulardetectorsprovider.dynamic duration=1.151943ms
INFO [07-05|16:16:44] Loading plugins...                       logger=plugin.store
ERROR[07-05|16:16:44] Failed to load external plugins          logger=plugin.sources error="failed to open plugins path"
ERROR[07-05|16:16:44] Could not register plugin                logger=plugins.registration pluginId=xychart error="plugin xychart is already registered"
ERROR[07-05|16:16:44] Could not initialize plugin              logger=plugins.initialization pluginId=xychart error="plugin xychart is already registered"
INFO [07-05|16:16:44] Plugins loaded                           logger=plugin.store count=54 duration=180.878188ms
INFO [07-05|16:16:44] Caching service is disabled              logger=caching.service
INFO [07-05|16:16:44] Query Service initialization             logger=query_data
INFO [07-05|16:16:44] Live Push Gateway initialization         logger=live.push_http
INFO [07-05|16:16:44] Applying new configuration to Alertmanager logger=ngalert.notifier.alertmanager org=1 configHash=d2c56faca6af2a5772ff4253222f7386
INFO [07-05|16:16:44] Running in alternative execution of Error/NoData mode logger=ngalert.state.manager
INFO [07-05|16:16:44] registering usage stat providers         logger=infra.usagestats.collector usageStatsProvidersLen=3
INFO [07-05|16:16:44] starting to provision alerting           logger=provisioning.alerting
INFO [07-05|16:16:44] finished to provision alerting           logger=provisioning.alerting
INFO [07-05|16:16:44] Warming state cache for startup          logger=ngalert.state.manager
INFO [07-05|16:16:44] State cache has been initialized         logger=ngalert.state.manager states=0 duration=728.117µs
ERROR[07-05|16:16:44] Failed to get renderer plugin sources    logger=renderer.manager error="failed to open plugins path"
INFO [07-05|16:16:44] HTTP Server Listen                       logger=http.server address=[::]:3000 protocol=http subUrl= socket=
INFO [07-05|16:16:44] Storage starting                         logger=grafanaStorageLogger
INFO [07-05|16:16:44] starting to provision dashboards         logger=provisioning.dashboard
INFO [07-05|16:16:44] finished to provision dashboards         logger=provisioning.dashboard
WARN [07-05|16:16:44] Scheduling and sending of reports disabled, SMTP is not configured and enabled. Configure SMTP to enable. logger=report
```

3 FUTURE SCOPE OF THE PROJECT

Since Prometheus, AlertManager and Grafana are not only used to observe metrics of Infrastructure, the next level of implementation would be to also consider instrumenting application code and scrape application specific metrics and widen the scope of infrastructure to, for example, the Kubernetes platform as Prometheus works natively with the Kubernetes deployed infrastructure and execute observability on tier 0 (highly critical) applications and infrastructure to use this platform to its most effective level.

4 REFERENCES

- https://prometheus.io/
- https://www.udemy.com/course/prometheus-course/
- https://techcrunch.com/