Set up Grafana , Prometheus and Blackbox Exporter

1. Task requirement:

Setting up Grafana, Prometheus, and Blackbox Exporter for monitoring.

2. Monitoring:

Monitoring is essential for ensuring the health and performance of your infrastructure, applications, and services. Grafana, Prometheus, and Blackbox Exporter are powerful tools that can be used to set up a comprehensive monitoring system. In this overview, we will outline the steps to set up and configure these tools for effective monitoring.

3. Environmental detail:

❖ **OS VERSION**

- Distributor ID: Ubuntu
- Description:	Ubuntu 22.04.3 LTS
- Release:	22.04
- Codename:	jammy

❖ **Podman VERSION**

- podman version 3.4.4

4. System configuration:

- CPU - 4
- Storage -16 GB

5. List of tools and technologies:

- Grafana
- Prometheus
- Black box exporter

Definition of tools:

1. Grafana

- Grafana is a popular open-source platform for creating, visualizing, and alerting on metrics and logs.
- It provides a user-friendly interface to build and customize dashboards.

2. Prometheus

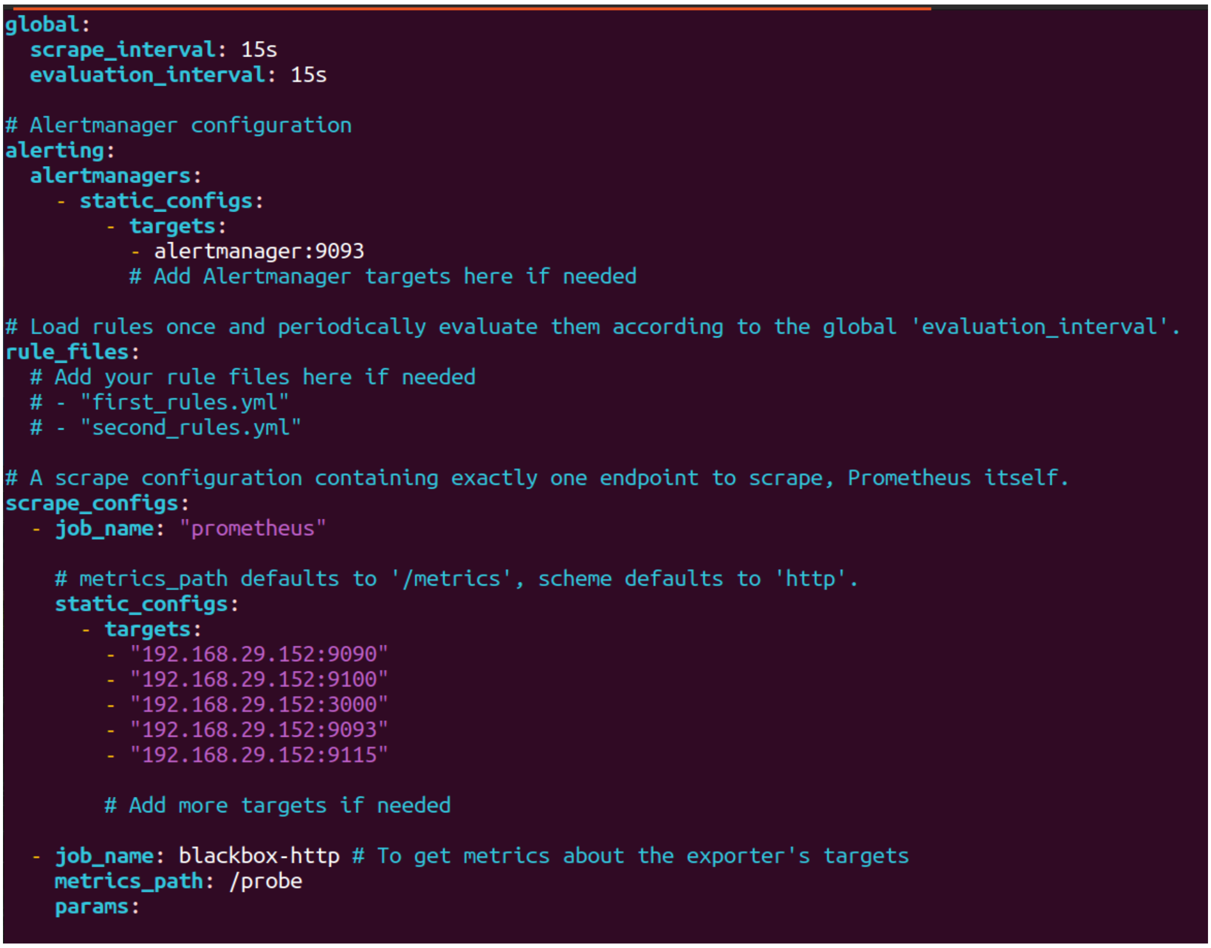
- Prometheus is an open-source monitoring and alerting toolkit designed for reliability and scalability.
- It collects and stores metrics from various sources, making them available for querying and alerting.

3. Blackbox Exporter

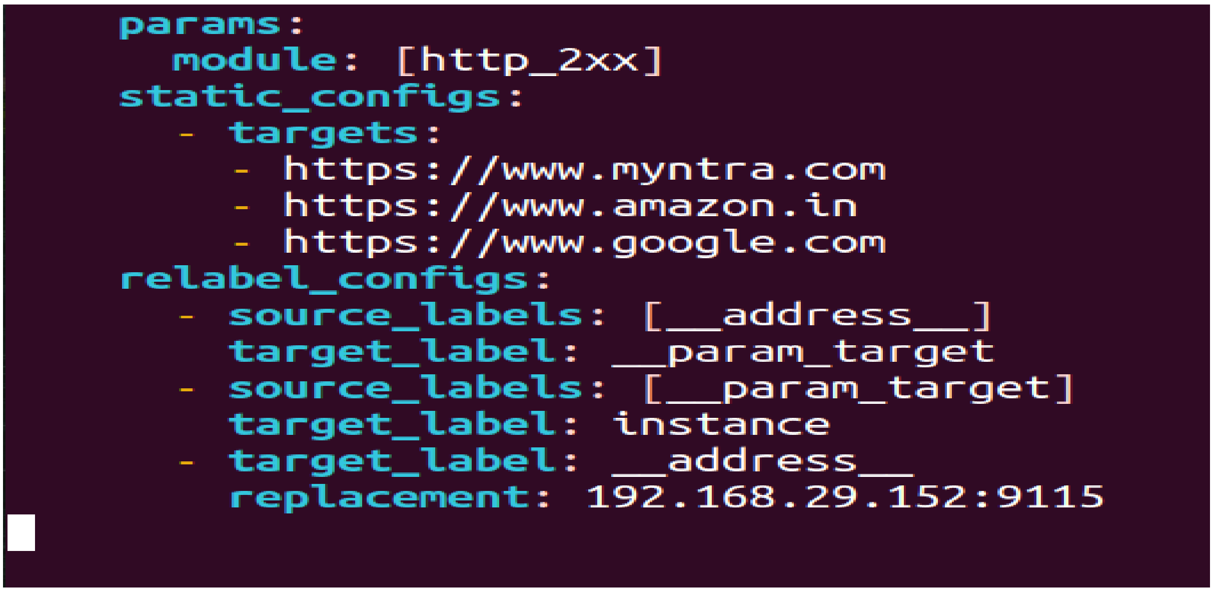
- Blackbox Exporter is a Prometheus exporter for probing endpoints over HTTP, HTTPS, DNS, TCP, and ICMP.
- It helps monitor the availability and response times of external services and endpoints.

Step 1. Create a prometheus.yml file and write this code .

vim prometheus.yml



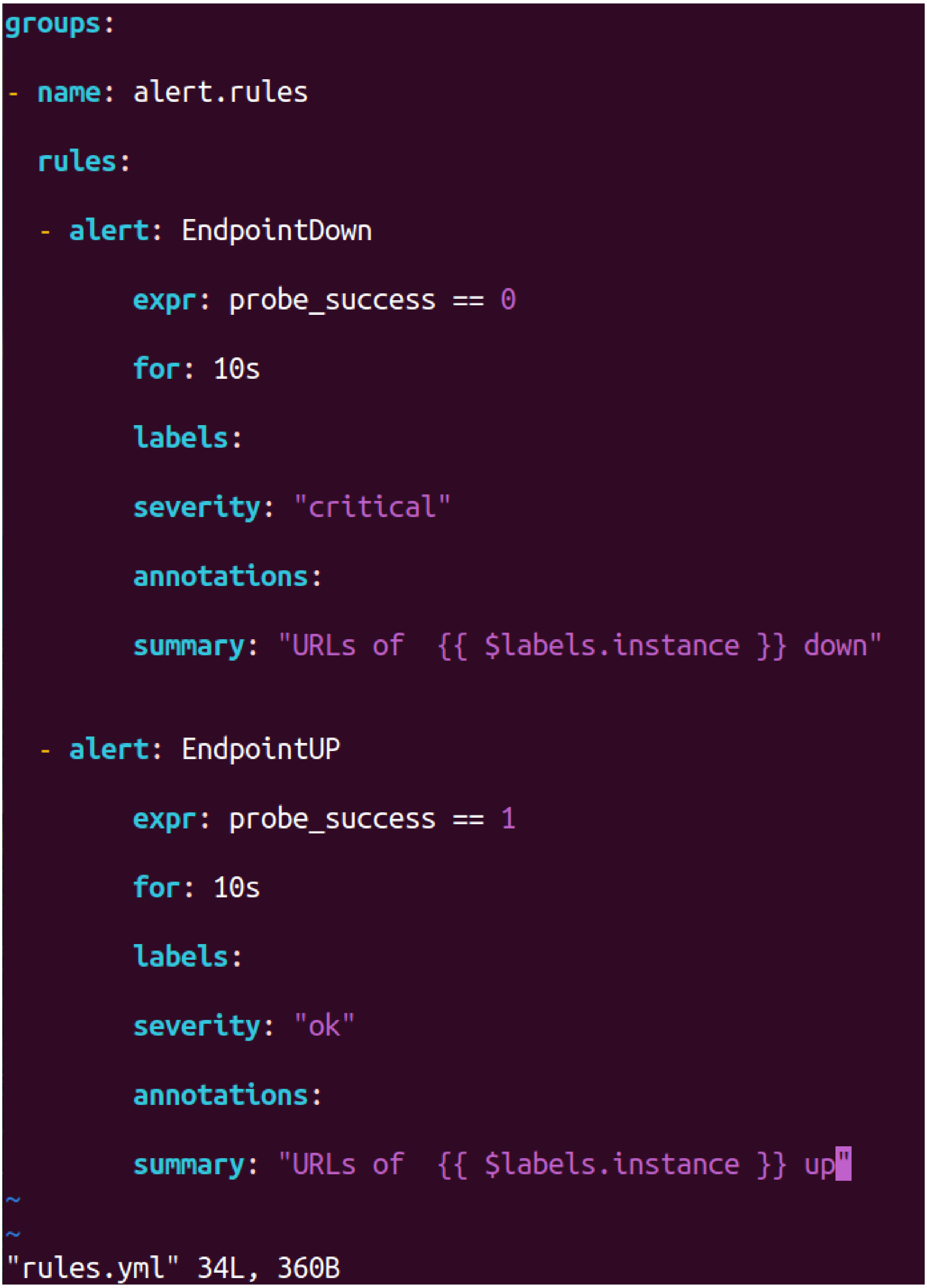
```
global:
  scrape_interval: 15s
  evaluation_interval: 15s

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          - alertmanager:9093
          # Add Alertmanager targets here if needed

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # Add your rule files here if needed
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape, Prometheus itself.
scrape_configs:
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics', scheme defaults to 'http'.
    static_configs:
      - targets:
        - "192.168.29.152:9090"
        - "192.168.29.152:9100"
        - "192.168.29.152:3000"
        - "192.168.29.152:9093"
        - "192.168.29.152:9115"

        # Add more targets if needed

  - job_name: blackbox-http # To get metrics about the exporter's targets
    metrics_path: /probe
    params:
```



```
    params:
      module: [http_2xx]
    static_configs:
      - targets:
        - https://www.myntra.com
        - https://www.amazon.in
        - https://www.google.com
    relabel_configs:
      - source_labels: [__address__]
        target_label: __param_target
      - source_labels: [__param_target]
        target_label: instance
      - target_label: __address__
        replacement: 192.168.29.152:9115
```

Network IP:192.168.29.152:9115
URL: https://www.google.com

Step 2. Create a rules.yml file and write this code .



```
groups:

- name: alert.rules

  rules:

  - alert: EndpointDown

        expr: probe_success == 0

        for: 10s

        labels:

        severity: "critical"

        annotations:

        summary: "URLs of  {{ $labels.instance }} down"


  - alert: EndpointUP

        expr: probe_success == 1

        for: 10s

        labels:

        severity: "ok"

        annotations:

        summary: "URLs of  {{ $labels.instance }} up"
~
~
"rules.yml" 34L, 360B
```

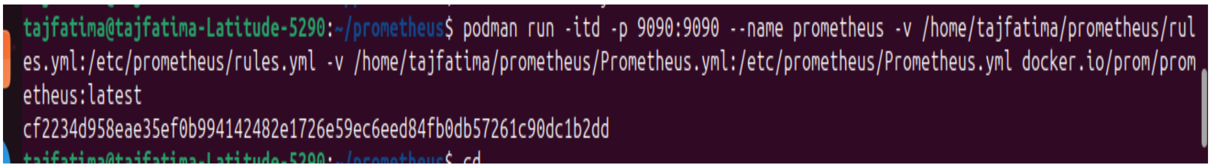
Step 3. Run the following command in your Grafana setup to start the Podman container:

A. Run the prometheus .

- podman run -d --name prometheus -p 9090:9090 -v /home/amit/prometheus/prometheus.yml:/etc/prometheus/prometheus.yml docker.io/prom/prometheus

`podman`: Container management tool.

1. `run`: Execute a container.
2. `-itd`: Interactive, TTY, Detached mode.
3. `-p 9090:9090`: Port mapping.
4. `--name prometheus`: Assigns a name.
5. `-v ...:/etc/prometheus/rules.yml`: Mounts rules.yml.
6. `-v ...:/etc/prometheus/prometheus.yml`: Mounts prometheus.yml.
7. `prom/prometheus:latest`: Docker image.

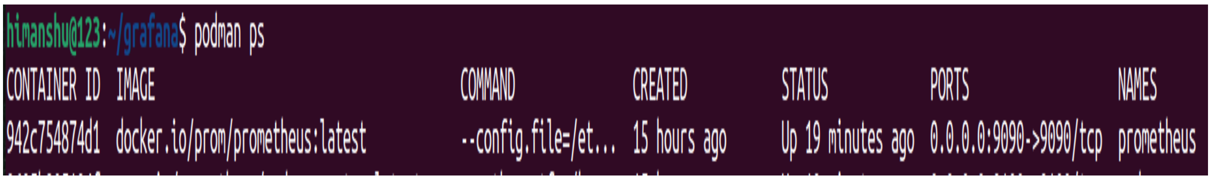


```
tajfatima@tajfatima-Latitude-5290:~/prometheus$ podman run -itd -p 9090:9090 --name prometheus -v /home/tajfatima/prometheus/rul
es.yml:/etc/prometheus/rules.yml -v /home/tajfatima/prometheus/Prometheus.yml:/etc/prometheus/Prometheus.yml docker.io/prom/prom
etheus:latest
cf2234d958eae35ef0b994142482e1726e59ec6eed84fb0db57261c90dc1b2dd
tajfatima@tajfatima-Latitude-5290:~/prometheus$ cd
```

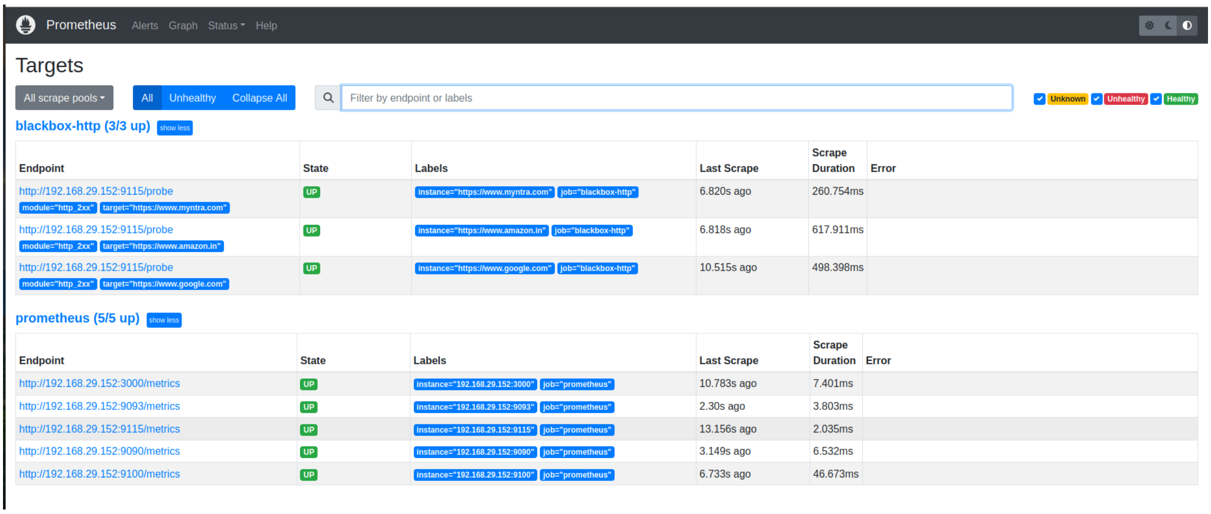
- Check :- podman ps

  1 .`podman`: Container management.

  2 .`ps`: List containers.



```
himanshu@123:~/grafana$ podman ps
CONTAINER ID  IMAGE                            COMMAND               CREATED       STATUS             PORTS                   NAMES
942c754874d1  docker.io/prom/prometheus:latest  --config.file=/et...  15 hours ago  Up 19 minutes ago  0.0.0.0:9090->9090/tcp  prometheus
```



Targets

blackbox-http (3/3 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.29.152:9115/probe module="http_2xx" target="https://www.myntra.com"	UP	instance="https://www.myntra.com" job="blackbox-http"	6.820s ago	260.754ms	
http://192.168.29.152:9115/probe module="http_2xx" target="https://www.amazon.in"	UP	instance="https://www.amazon.in" job="blackbox-http"	6.818s ago	617.911ms	
http://192.168.29.152:9115/probe module="http_2xx" target="https://www.google.com"	UP	instance="https://www.google.com" job="blackbox-http"	10.515s ago	498.398ms	

prometheus (5/5 up)

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.29.152:3000/metrics	UP	instance="192.168.29.152:3000" job="prometheus"	10.783s ago	7.401ms	
http://192.168.29.152:9093/metrics	UP	instance="192.168.29.152:9093" job="prometheus"	2.30s ago	3.803ms	
http://192.168.29.152:9115/metrics	UP	instance="192.168.29.152:9115" job="prometheus"	13.156s ago	2.035ms	
http://192.168.29.152:9090/metrics	UP	instance="192.168.29.152:9090" job="prometheus"	3.149s ago	6.532ms	
http://192.168.29.152:9100/metrics	UP	instance="192.168.29.152:9100" job="prometheus"	6.733s ago	46.673ms	

B. Run grafana .

- podman run -itd --name=grafana -p 3000:3000 docker.io/grafana/grafana:8.3.2

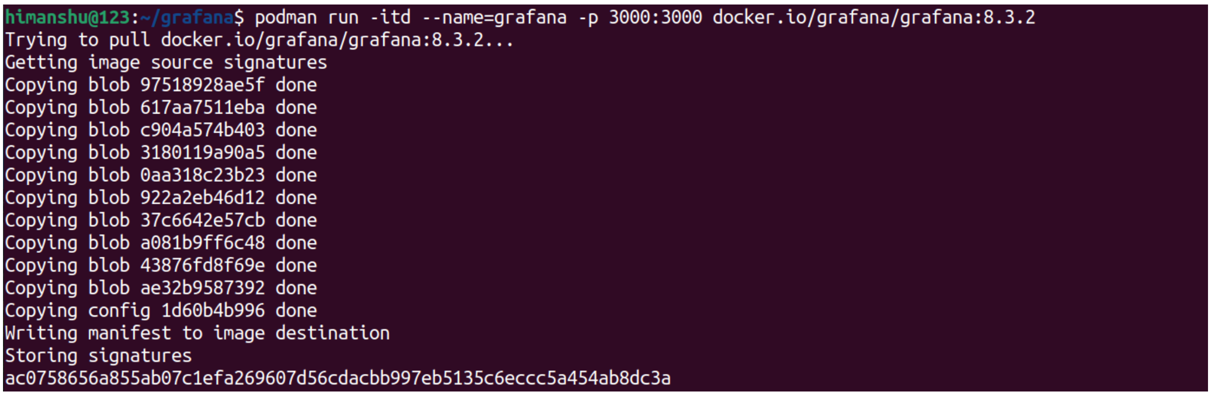
1. `podman`: The command itself, indicating that we are using Podman, a container management tool similar to Docker.
2. `run`: Instructs Podman to run a container based on an image.
3. `-itd`: These are flags that modify how the container is run:
   - `-i`: Keep STDIN open (interactive).
   - `-t`: Allocate a pseudo-TTY (terminal).
   - `-d`: Run the container in the background (detached).
4. `--name=grafana`: Assigns the name "grafana" to the container, making it easier to manage and reference later.
5. `-p 3000:3000`: Specifies port mapping, where traffic on port 3000 of the host is forwarded to port 3000 inside the container. This is commonly used for accessing services running in the container.

6. `docker.io/grafana/grafana:8.3.2`: Specifies the container image to use, in this case, "grafana/grafana" version 8.3.2, retrieved from the Docker Hub repository.

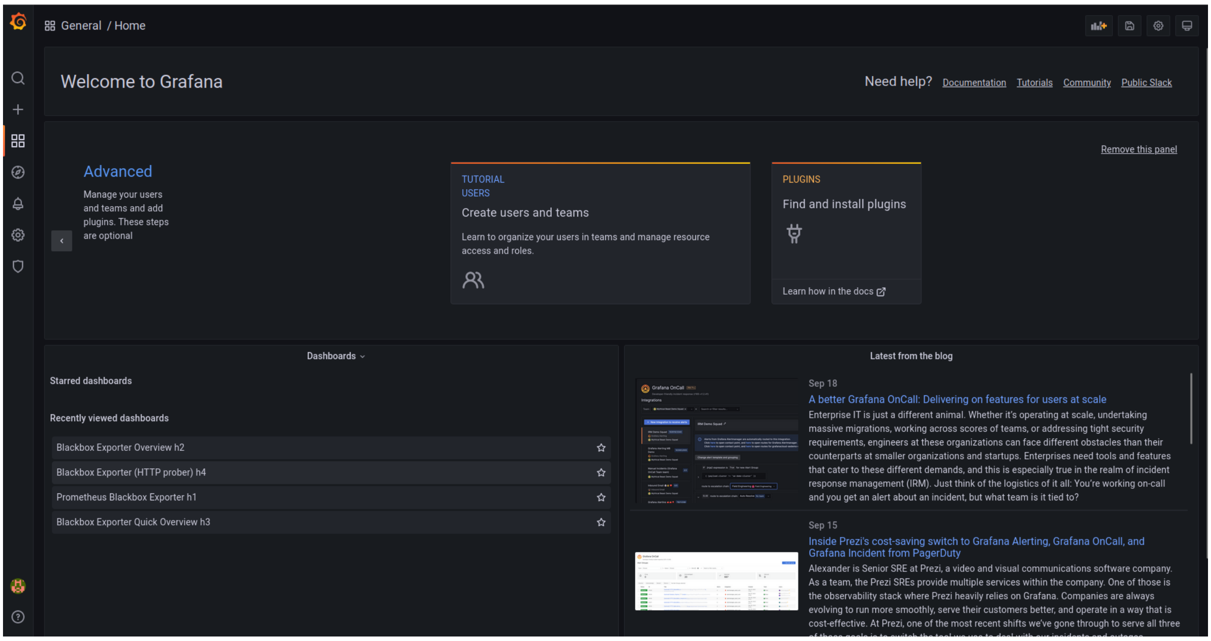
- Check :- podman ps

  1. `podman`: Container management.

  2. `ps`: List containers.



```
himanshu@123:~/grafana$ podman run -itd --name=grafana -p 3000:3000 docker.io/grafana/grafana:8.3.2
Trying to pull docker.io/grafana/grafana:8.3.2...
Getting image source signatures
Copying blob 97518928ae5f done
Copying blob 617aa7511eba done
Copying blob c904a574b403 done
Copying blob 3180119a90a5 done
Copying blob 0aa318c23b23 done
Copying blob 922a2eb46d12 done
Copying blob 37c6642e57cb done
Copying blob a081b9ff6c48 done
Copying blob 43876fd8f69e done
Copying blob ae32b9587392 done
Copying config 1d60b4b996 done
Writing manifest to image destination
Storing signatures
ac0758656a855ab07c1efa269607d56cdacbb997eb5135c6eccc5a454ab8dc3a
```



General / Home
Welcome to Grafana
Need help? Documentation Tutorials Community Public Slack
Remove this panel
Advanced
TUTORIAL
USERS
Create users and teams
PLUGINS
Find and install plugins
Learn how in the docs
Dashboards
Starred dashboards
Recently viewed dashboards
Blackbox Exporter Overview h2
Blackbox Exporter (HTTP prober) h4
Prometheus Blackbox Exporter h1
Blackbox Exporter Quick Overview h3
Latest from the blog

Manage your users and teams and add plugins. These steps are optional

Learn to organize your users in teams and manage resource access and roles.

Sep 18

A better Grafana OnCall: Delivering on features for users at scale

Enterprise IT is just a different animal. Whether it's operating at scale, undertaking massive migrations, working across scores of teams, or addressing tight security requirements, engineers at these organizations can face different obstacles than their counterparts at smaller organizations and startups. Enterprises need tools and features that cater to these different demands, and this is especially true in the realm of incident response management (IRM). Just think of the logistics of it all: You're working on-call and you get an alert about an incident, but what team is it tied to?

Sep 15

Inside Prezi's cost-saving switch to Grafana Alerting, Grafana OnCall, and Grafana Incident from PagerDuty

Alexander is Senior SRE at Prezi, a video and visual communications software company. As a team, the Prezi SREs provide multiple services within the company. One of those is the observability stack where Prezi heavily relies on Grafana. Companies are always evolving to run more smoothly, serve their customers better, and operate in a way that is cost-effective. At Prezi, one of the most recent shifts we've gone through to serve all three

C. Run blackbox exporter .

- podman run -d --name black -p 9115:9115 bitnami/blackbox-exporter:latest

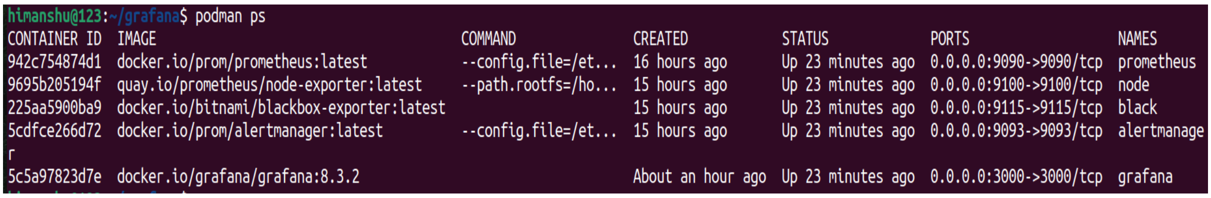
1. `podman`: Container management tool.

2. `run`: Start a container.
3. `-d`: Run in detached mode.
4. `--name black`: Assign the name "black" to the container.
5. `-p 9115:9115`: Map port 9115 from host to container.
6. `bitnami/blackbox-exporter:latest`: Docker image to run.

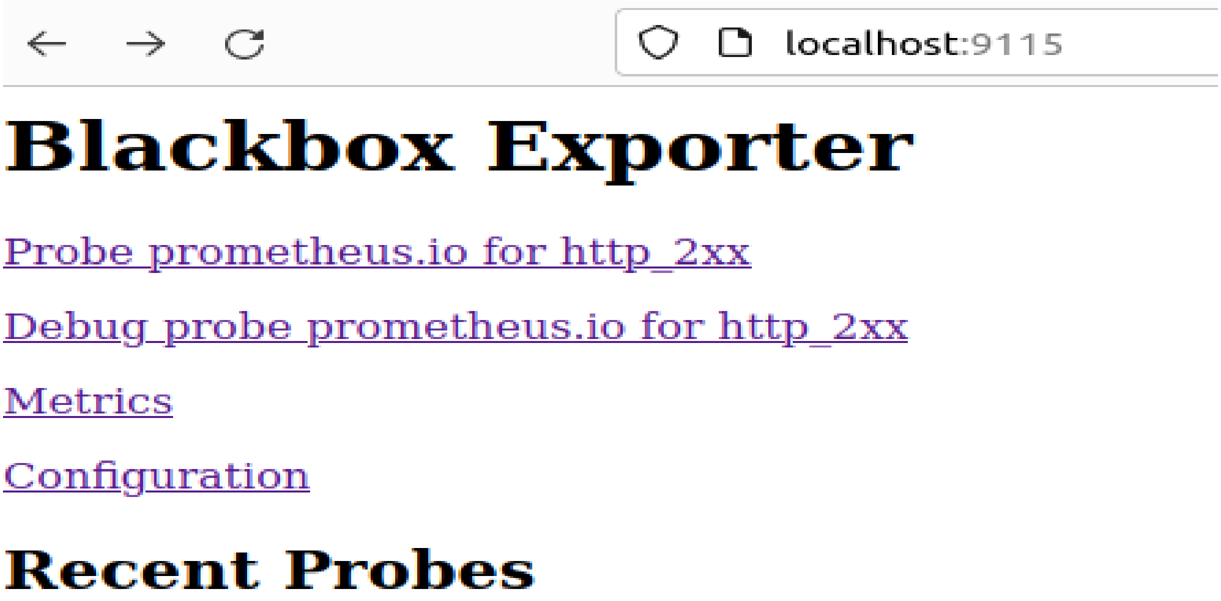
- Check :- podman ps

  1 . `podman`: Container management.

  2 . `ps`: List containers.



```
himanshu@123:~/grafana$ podman ps
CONTAINER ID  IMAGE                                    COMMAND               CREATED             STATUS              PORTS                   NAMES
942c754874d1  docker.io/prom/prometheus:latest         --config.file=/et...  16 hours ago        Up 23 minutes ago   0.0.0.0:9090->9090/tcp  prometheus
9695b205194f  quay.io/prometheus/node-exporter:latest  --path.rootfs=/ho...  15 hours ago        Up 23 minutes ago   0.0.0.0:9100->9100/tcp  node
225aa5900ba9  docker.io/bitnami/blackbox-exporter:latest                     15 hours ago        Up 23 minutes ago   0.0.0.0:9115->9115/tcp  black
5cdfce266d72  docker.io/prom/alertmanager:latest       --config.file=/et...  15 hours ago        Up 23 minutes ago   0.0.0.0:9093->9093/tcp  alertmanage
r
5c5a97823d7e  docker.io/grafana/grafana:8.3.2                                About an hour ago   Up 23 minutes ago   0.0.0.0:3000->3000/tcp  grafana
```



Blackbox Exporter

Probe prometheus.io for http_2xx

Debug probe prometheus.io for http_2xx

Metrics

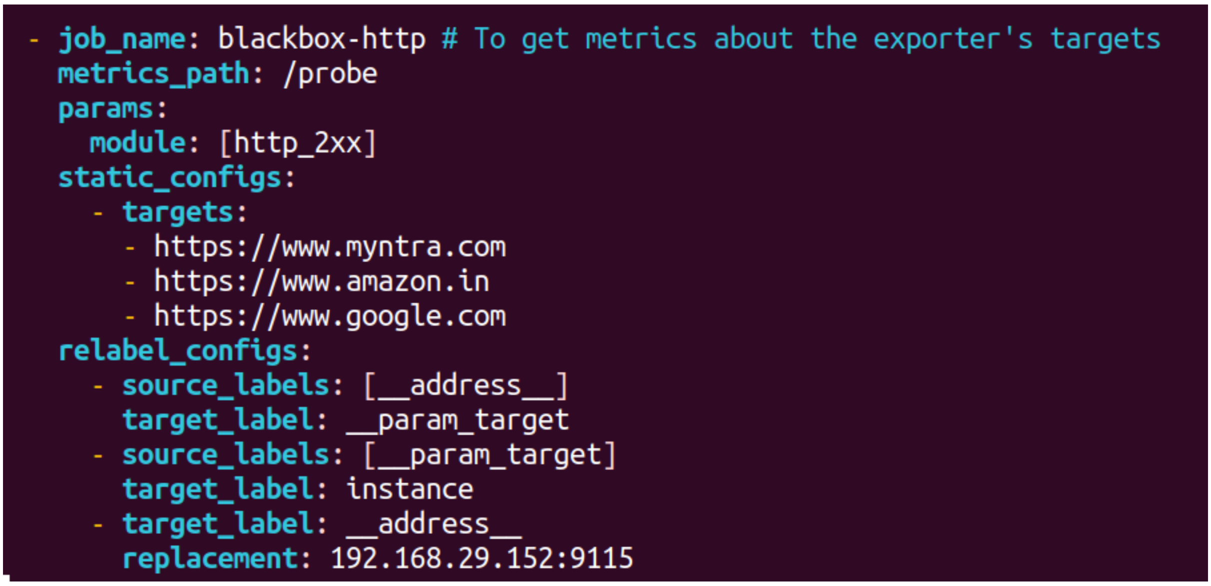
Configuration

Recent Probes

Step 4. How do you monitor any URL and how do you configure it?

Our team has monitored Google.com, Amazon, and Myntra on Grafana.

A. We need to first put the URLs of all three in the prometheus.yml file.



```
- job_name: blackbox-http # To get metrics about the exporter's targets
  metrics_path: /probe
  params:
    module: [http_2xx]
  static_configs:
    - targets:
      - https://www.myntra.com
      - https://www.amazon.in
      - https://www.google.com
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: 192.168.29.152:9115
```

- You will need to add your own IP address in your Prometheus.yml file.

Restart the container .

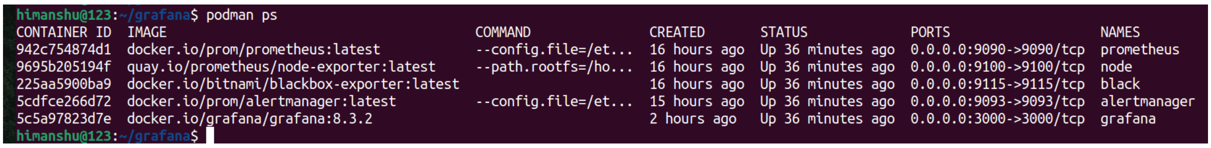
- podman restart <container-id>

1. podman: The command-line tool used for managing containers. It's similar to Docker.
2. restart: The action to stop and then start a container.
3. <container-id>: Replace this with the actual ID or name of the container you want to restart.

- podman ps

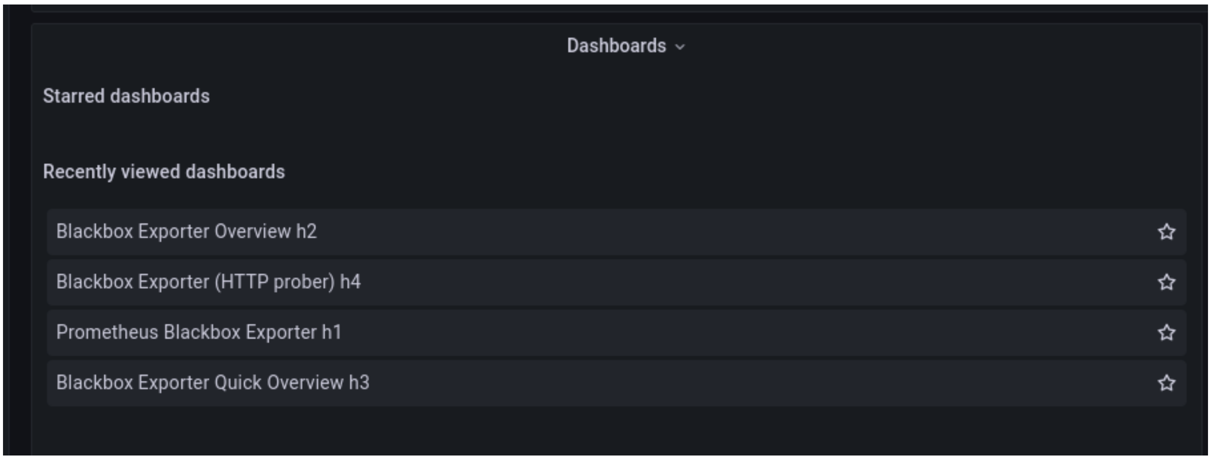
1 . podman: Container management.

2 . ps: List containers.



```
himanshu@123:~/grafana$ podman ps
CONTAINER ID  IMAGE                                     COMMAND               CREATED       STATUS             PORTS                   NAMES
942c754874d1  docker.io/prom/prometheus:latest          --config.file=/et...  16 hours ago  Up 36 minutes ago  0.0.0.0:9090->9090/tcp  prometheus
9695b205194f  quay.io/prometheus/node-exporter:latest   --path.rootfs=/ho...  16 hours ago  Up 36 minutes ago  0.0.0.0:9100->9100/tcp  node
225aa5900ba9  docker.io/bitnami/blackbox-exporter:latest                      16 hours ago  Up 36 minutes ago  0.0.0.0:9115->9115/tcp  black
5cdfce266d72  docker.io/prom/alertmanager:latest        --config.file=/et...  15 hours ago  Up 36 minutes ago  0.0.0.0:9093->9093/tcp  alertmanager
5c5a97823d7e  docker.io/grafana/grafana:8.3.2                                 2 hours ago   Up 36 minutes ago  0.0.0.0:3000->3000/tcp  grafana
himanshu@123:~/grafana$
```

Create a dashboard



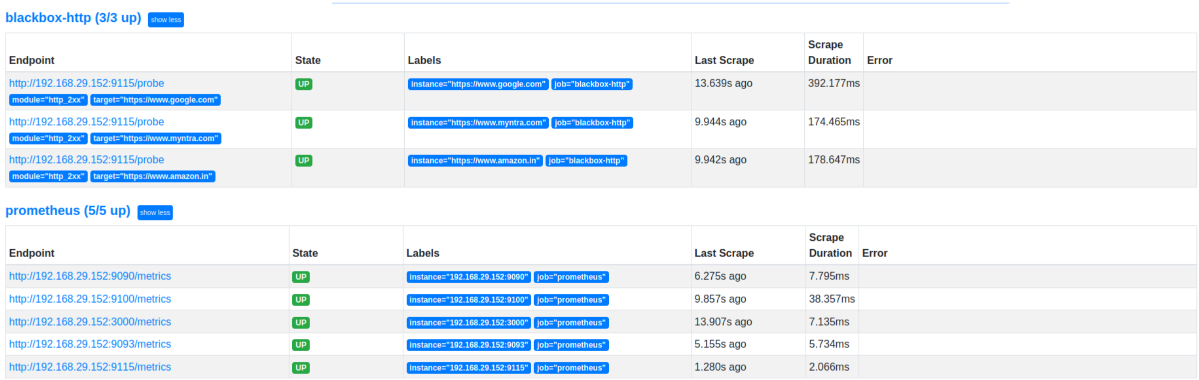
Dashboards ⌄

Starred dashboards

Recently viewed dashboards

- Blackbox Exporter Overview h2 ☆
- Blackbox Exporter (HTTP prober) h4 ☆
- Prometheus Blackbox Exporter h1 ☆
- Blackbox Exporter Quick Overview h3 ☆

Check prometheus .



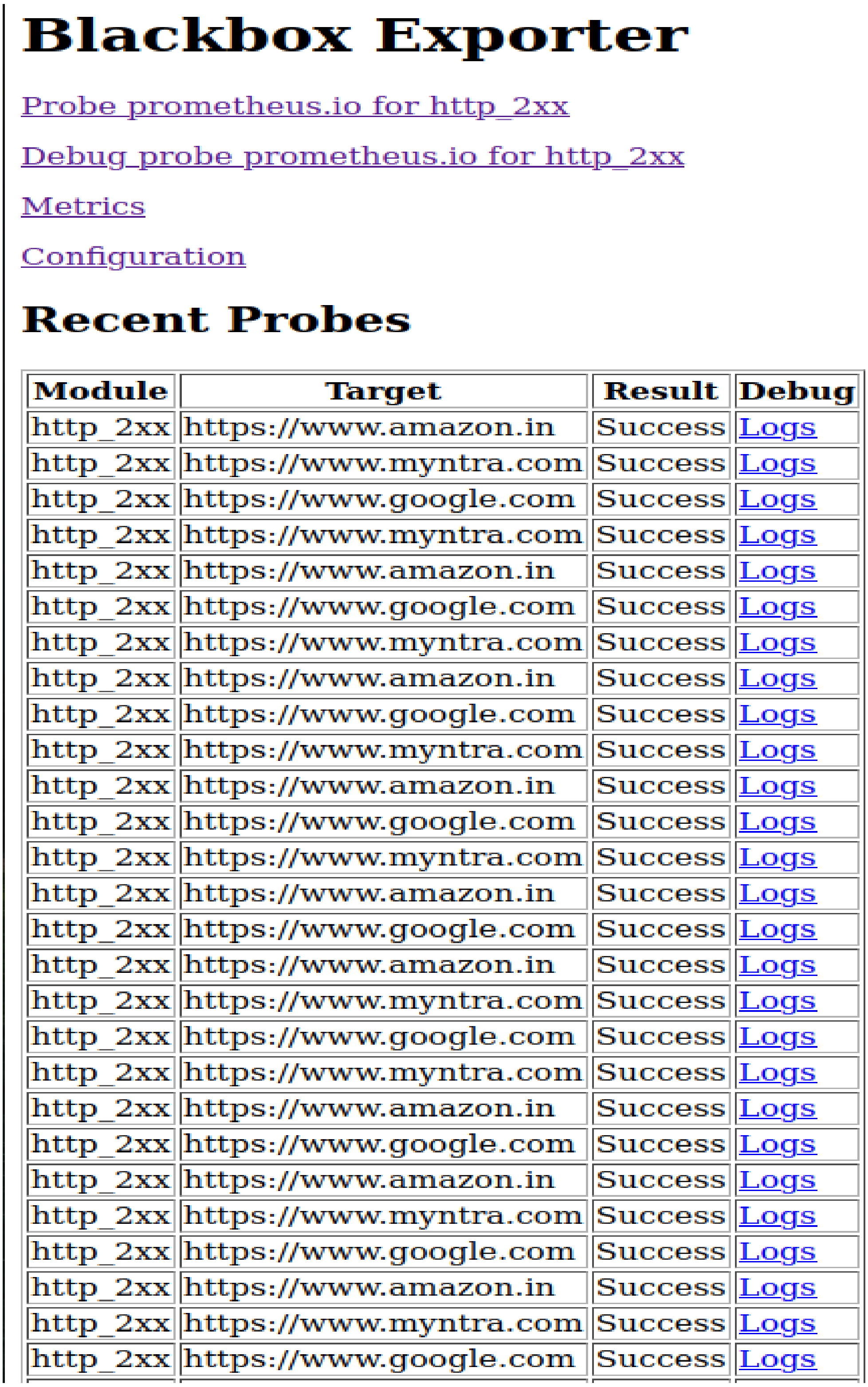
blackbox-http (3/3 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.29.152:9115/probe module="http_2xx" target="https://www.google.com"	UP	instance="https://www.google.com" job="blackbox-http"	13.639s ago	392.177ms	
http://192.168.29.152:9115/probe module="http_2xx" target="https://www.myntra.com"	UP	instance="https://www.myntra.com" job="blackbox-http"	9.944s ago	174.465ms	
http://192.168.29.152:9115/probe module="http_2xx" target="https://www.amazon.in"	UP	instance="https://www.amazon.in" job="blackbox-http"	9.942s ago	178.647ms	

prometheus (5/5 up) show less

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
http://192.168.29.152:9090/metrics	UP	instance="192.168.29.152:9090" job="prometheus"	6.275s ago	7.795ms	
http://192.168.29.152:9100/metrics	UP	instance="192.168.29.152:9100" job="prometheus"	9.857s ago	38.357ms	
http://192.168.29.152:3000/metrics	UP	instance="192.168.29.152:3000" job="prometheus"	13.907s ago	7.135ms	
http://192.168.29.152:9093/metrics	UP	instance="192.168.29.152:9093" job="prometheus"	5.155s ago	5.734ms	
http://192.168.29.152:9115/metrics	UP	instance="192.168.29.152:9115" job="prometheus"	1.280s ago	2.066ms	

Check blackbox exporter matrics .



Blackbox Exporter

Probe prometheus.io for http_2xx

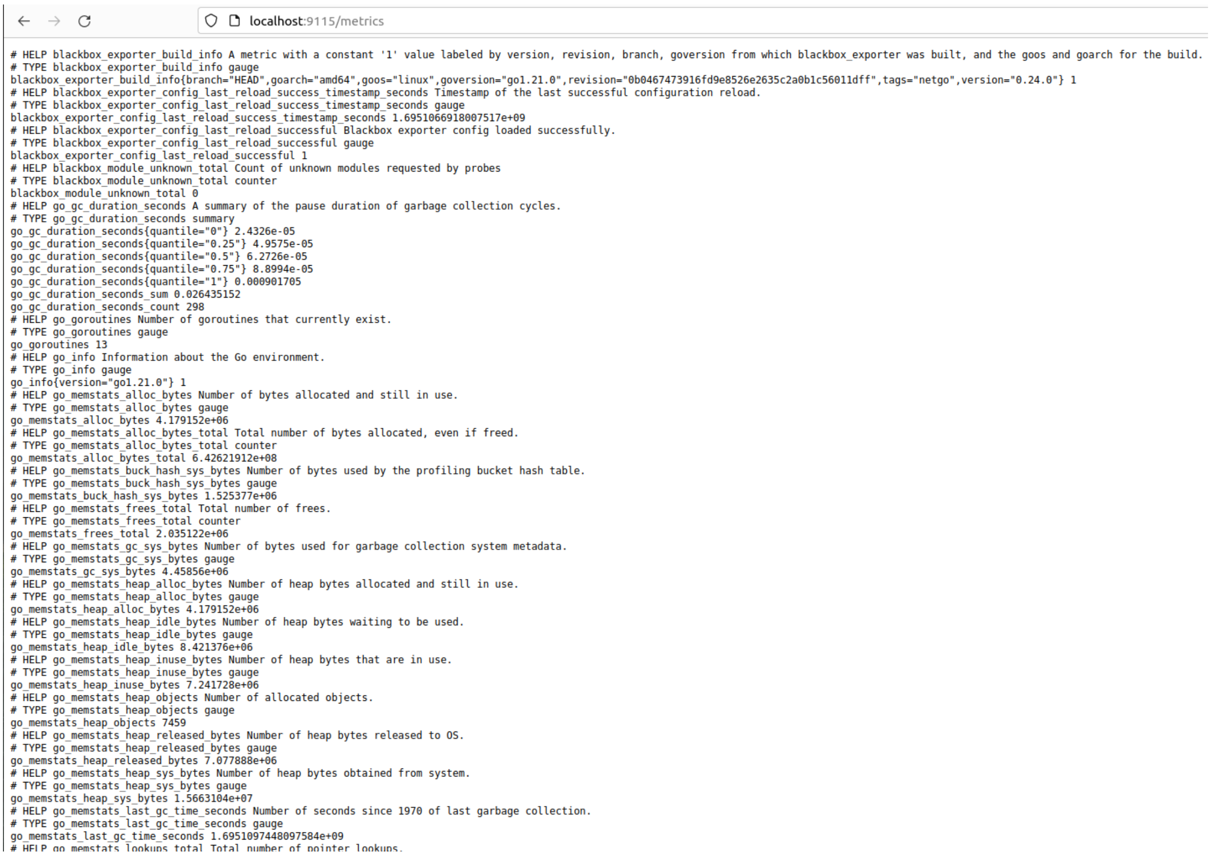
Debug probe prometheus.io for http_2xx

Metrics

Configuration

Recent Probes

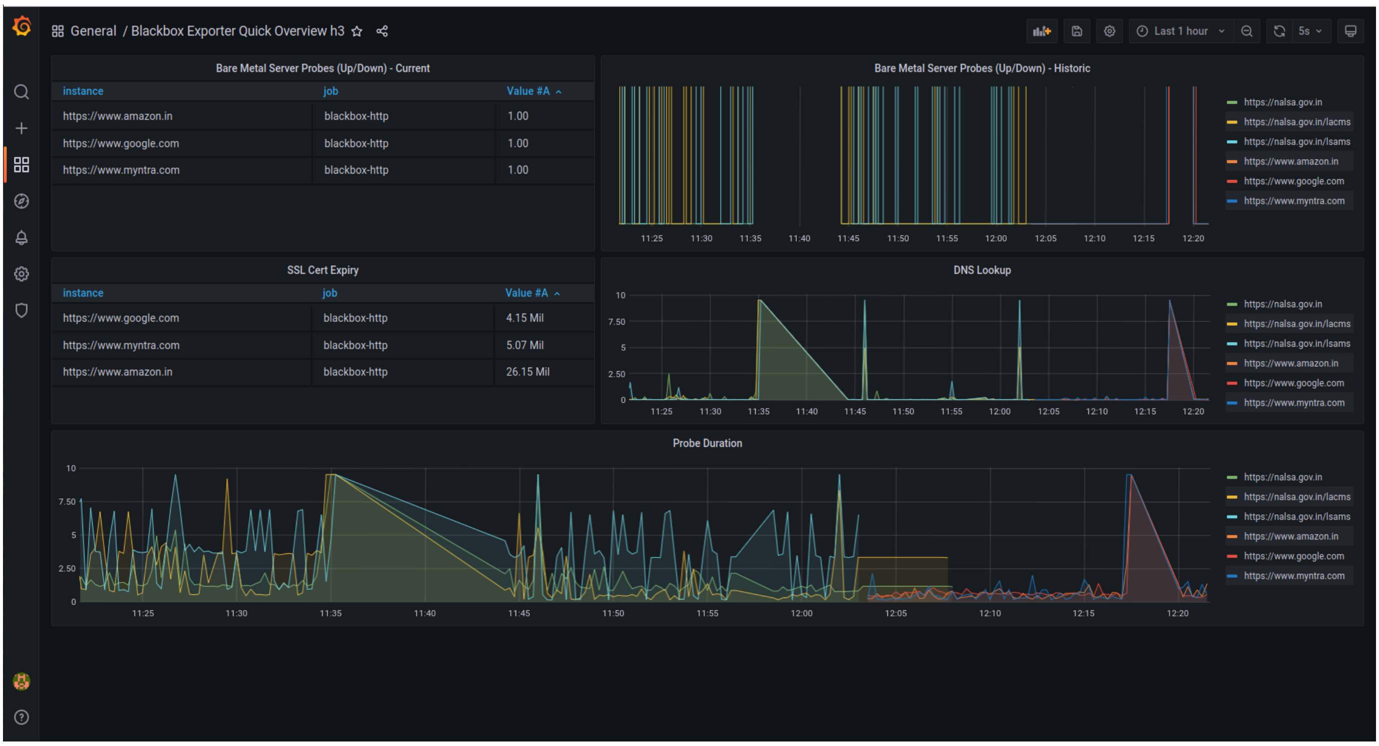
Module	Target	Result	Debug
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.google.com	Success	Logs
http_2xx	https://www.amazon.in	Success	Logs
http_2xx	https://www.myntra.com	Success	Logs
http_2xx	https://www.google.com	Success	Logs



localhost:9115/metrics

```
# HELP blackbox_exporter_build_info A metric with a constant '1' value labeled by version, revision, branch, goversion from which blackbox_exporter was built, and the goos and goarch for the build.
# TYPE blackbox_exporter_build_info gauge
blackbox_exporter_build_info{branch="HEAD",goarch="amd64",goos="linux",goversion="go1.21.0",revision="0b0467473916fd9e8526e2635c2a0b1c56011dff",tags="netgo",version="0.24.0"} 1
# HELP blackbox_exporter_config_last_reload_success_timestamp_seconds Timestamp of the last successful configuration reload.
# TYPE blackbox_exporter_config_last_reload_success_timestamp_seconds gauge
blackbox_exporter_config_last_reload_success_timestamp_seconds 1.6951066918007517e+09
# HELP blackbox_exporter_config_last_reload_successful Blackbox exporter config loaded successfully.
# TYPE blackbox_exporter_config_last_reload_successful gauge
blackbox_exporter_config_last_reload_successful 1
# HELP blackbox_module_unknown_total Count of unknown modules requested by probes
# TYPE blackbox_module_unknown_total counter
blackbox_module_unknown_total 0
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 2.4326e-05
go_gc_duration_seconds{quantile="0.25"} 4.9575e-05
go_gc_duration_seconds{quantile="0.5"} 6.2726e-05
go_gc_duration_seconds{quantile="0.75"} 8.8994e-05
go_gc_duration_seconds{quantile="1"} 0.000901705
go_gc_duration_seconds_sum 0.026435152
go_gc_duration_seconds_count 298
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 13
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.21.0"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 4.179152e+06
# HELP go_memstats_alloc_bytes_total Total number of bytes allocated, even if freed.
# TYPE go_memstats_alloc_bytes_total counter
go_memstats_alloc_bytes_total 6.42621912e+08
# HELP go_memstats_buck_hash_sys_bytes Number of bytes used by the profiling bucket hash table.
# TYPE go_memstats_buck_hash_sys_bytes gauge
go_memstats_buck_hash_sys_bytes 1.525377e+06
# HELP go_memstats_frees_total Total number of frees.
# TYPE go_memstats_frees_total counter
go_memstats_frees_total 2.035122e+06
# HELP go_memstats_gc_sys_bytes Number of bytes used for garbage collection system metadata.
# TYPE go_memstats_gc_sys_bytes gauge
go_memstats_gc_sys_bytes 4.45856e+06
# HELP go_memstats_heap_alloc_bytes Number of heap bytes allocated and still in use.
# TYPE go_memstats_heap_alloc_bytes gauge
go_memstats_heap_alloc_bytes 4.179152e+06
# HELP go_memstats_heap_idle_bytes Number of heap bytes waiting to be used.
# TYPE go_memstats_heap_idle_bytes gauge
go_memstats_heap_idle_bytes 8.421376e+06
# HELP go_memstats_heap_inuse_bytes Number of heap bytes that are in use.
# TYPE go_memstats_heap_inuse_bytes gauge
go_memstats_heap_inuse_bytes 7.241728e+06
# HELP go_memstats_heap_objects Number of allocated objects.
# TYPE go_memstats_heap_objects gauge
go_memstats_heap_objects 7459
# HELP go_memstats_heap_released_bytes Number of heap bytes released to OS.
# TYPE go_memstats_heap_released_bytes gauge
go_memstats_heap_released_bytes 7.077888e+06
# HELP go_memstats_heap_sys_bytes Number of heap bytes obtained from system.
# TYPE go_memstats_heap_sys_bytes gauge
go_memstats_heap_sys_bytes 1.5663104e+07
# HELP go_memstats_last_gc_time_seconds Number of seconds since 1970 of last garbage collection.
# TYPE go_memstats_last_gc_time_seconds gauge
go_memstats_last_gc_time_seconds 1.6951097448097584e+09
# HELP go_memstats_lookups_total Total number of pointer lookups.
```

Step 5. Now let's start monitoring in Grafana .



Bare Metal Server Probes (Up/Down) - Current

instance	job	Value #A
https://www.amazon.in	blackbox-http	1.00
https://www.google.com	blackbox-http	1.00
https://www.myntra.com	blackbox-http	1.00

SSL Cert Expiry

instance	job	Value #A
https://www.google.com	blackbox-http	4.15 Mil
https://www.myntra.com	blackbox-http	5.07 Mil
https://www.amazon.in	blackbox-http	26.15 Mil

Bare Metal Server Probes (Up/Down) - Historic

DNS Lookup

Probe Duration