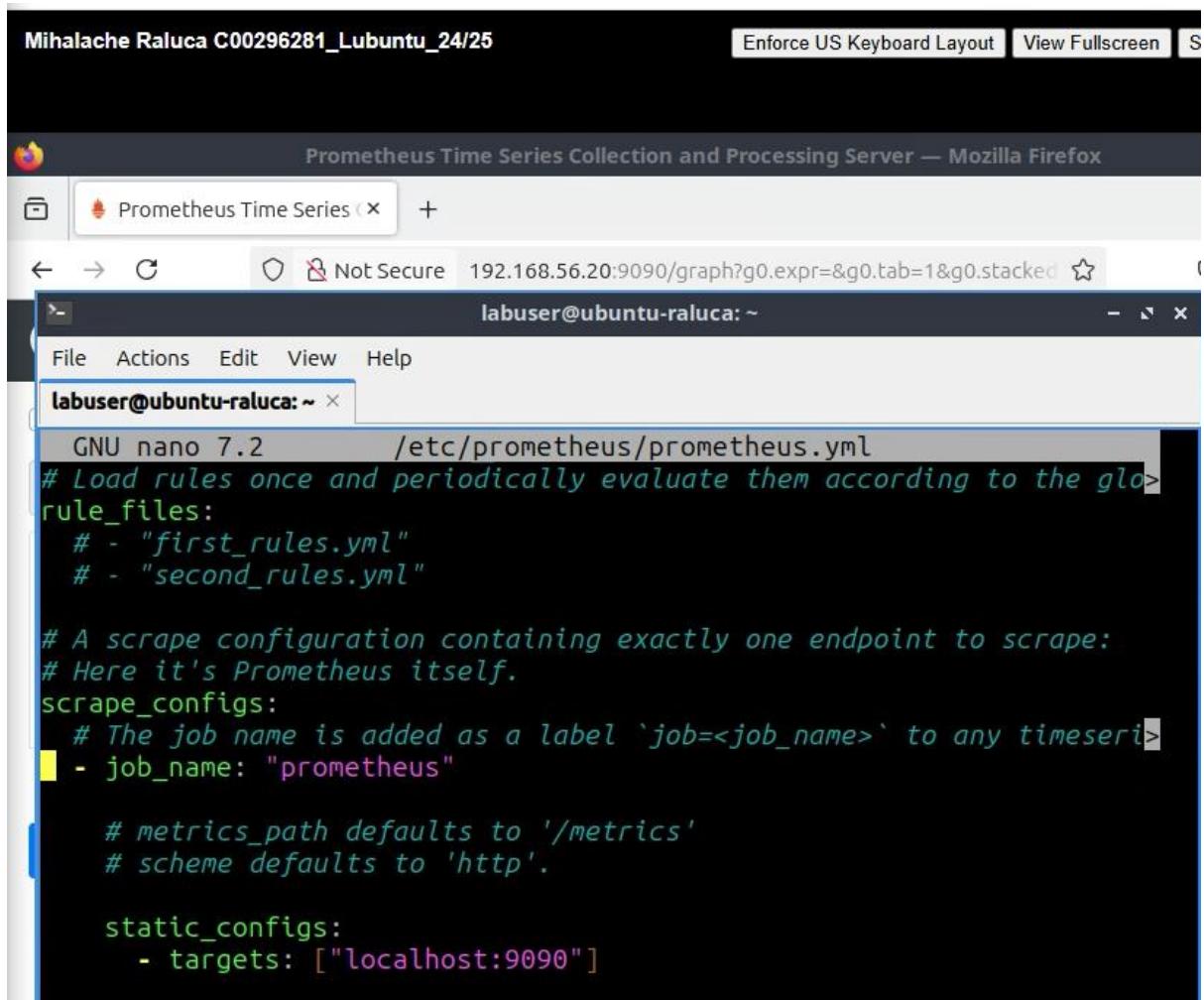


# System Infrastructure and Security Project Report

In this project, I set up a basic security and monitoring system using tools like Prometheus, Grafana, and Fail2Ban on the Ubuntu server. The goal was to monitor the system, track important security data, and protect it from common threats like repeated failed logins. Prometheus was used to collect and alert on system data, Grafana was used to display this data in dashboard, and Fail2Ban helped block suspicious login attempts. This project helped me understand how to improve system security and keep track of what's happening on a server.

Prometheus was installed on the Ubuntu server by downloading the binary file, extracting it and setting it up as a systemd service. The main configuration file, **prometheus.yml**, was edited to set the global scrape interval and define Prometheus itself as a scrape target. This allowed Prometheus to collect its own metrics for monitoring. After starting the service, Prometheus was accessible through a web browser at <http://192.168.56.20:9090>.



Mihalache Raluca C00296281\_Lubuntu\_24/25      Enforce US Keyboard Layout      View Fullscreen      S

Prometheus Time Series Collection and Processing Server — Mozilla Firefox

Prometheus Time Series × +

Not Secure 192.168.56.20:9090/graph?g0.expr=&g0.tab=1&g0.stacked ☆

labuser@ubuntu-raluca: ~

File Actions Edit View Help

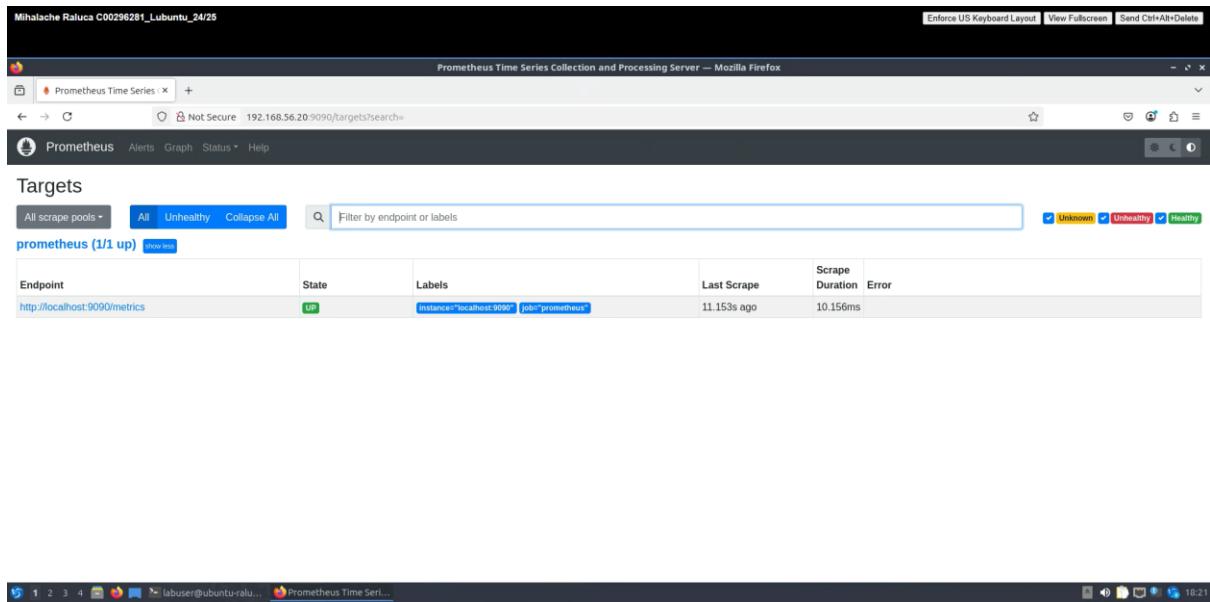
GNU nano 7.2 /etc/prometheus/prometheus.yml

```
# Load rules once and periodically evaluate them according to the global evaluation interval.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape.
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped by this configuration.
  - job_name: "prometheus"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

    static_configs:
      - targets: ["localhost:9090"]
```



Node Exporter was installed to allow Prometheus to collect detailed system metrics such as CPU usage, memory and disk space. To install it, I began by navigating to the **/tmp** directory and using **wget** to download the latest Node Exporter tar.gz file from the official Prometheus release page. The file was extracted using the **tar** command, and the binary file was moved to **/usr/local/bin**. A system user named **node\_exporter** was then created using **sudo useradd -rs /bin/false node\_exporter**. To run Node Exporter continuously in the background, a new systemd service file was created at **/etc/systemd/system/node\_exporter.service**.

Mihalache Raluca C00296281\_Lubuntu\_24/25      Enforce US Keyboard Layout | View Fullscreen | S

Prometheus Time Series Collection and Processing Server — Mozilla Firefox

Prometheus Time Series (x) +

Not Secure 192.168.56.20:9090/targets?search=

labuser@ubuntu-raluca: ~

File Actions Edit View Help

labuser@ubuntu-raluca: ~

```
GNU nano 7.2 /etc/systemd/system/node_exporter.service *
[Unit]
Description=Node Exporter
After=network.target

[Service]
User=node_exporter
Group=node_exporter
Type=simple
ExecStart=/usr/local/bin/node_exporter

[Install]
WantedBy=multi-user.target
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute  
 ^X Exit ^R Read File ^A Replace ^U Paste ^J Justify

After creating the file, the service was enabled, and once running on port **9100**, Node Exporter was added to the Prometheus configuration file as a target, allowing Prometheus to begin collecting system metrics.

Mihalache Raluca C00296281\_Lubuntu\_24/25      Enforce US Keyboard Layout | View Fullscreen | Send Ctrl+Alt+Delete

Prometheus Time Series Collection and Processing Server — Mozilla Firefox

Prometheus Time Series (x) +

Not Secure 192.168.56.20:9090/targets?search=

Targets

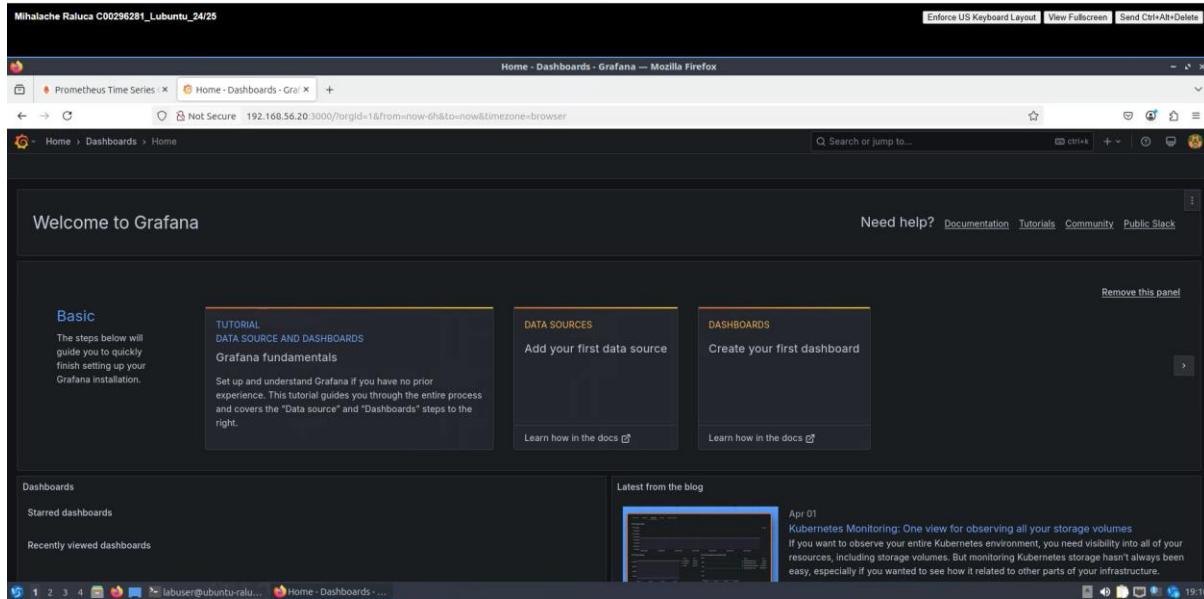
All scrape pools: All Unhealthy Collapse All Filter by endpoint or labels   Unknown  Unhealthy  Healthy

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="#">http://localhost:9090/metrics</a>	UP	instance="localhost:9090" job="prometheus"	4.843s ago	9.349ms	

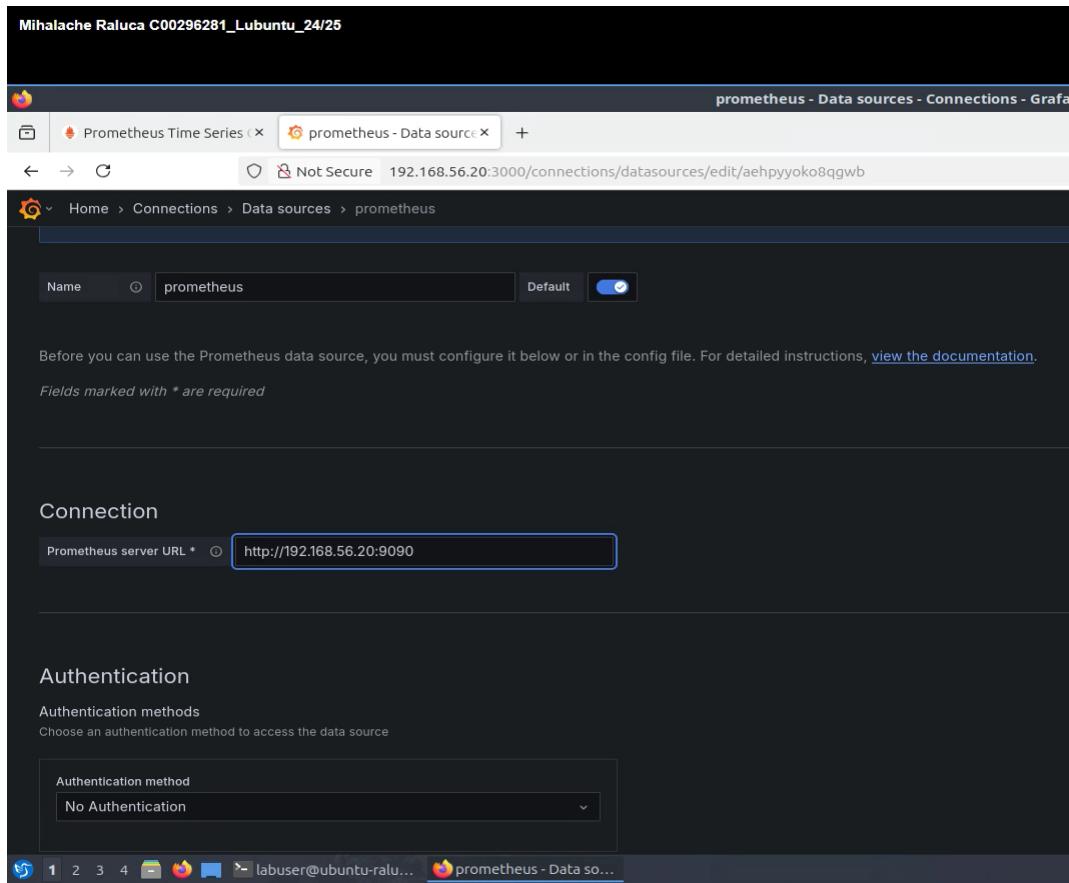


Grafana was installed on the Ubuntu server to provide dashboards for visualizing system and security metrics. The installation began by adding the Grafana GPG key and repository using the terminal. After updating the package list with **sudo apt update**, Grafana was

installed using **sudo apt install grafana**. The service was then started and enabled with **sudo systemctl start grafana-server** and **sudo systemctl enable grafana-server**. Once running, Grafana was accessible through a web browser at <http://192.168.56.20:3000>. The default login credentials were **admin** for both username and password. After logging in, Grafana was ready to be connected to Prometheus as a data source.



To connect Grafana with Prometheus, Prometheus was added as a data source in Grafana. This was done by navigating to the Data Sources section and selecting Prometheus. In the configuration page, the URL was set to <http://192.168.56.20:9090>, which is the default address where Prometheus runs. After entering the URL, the connection was tested by pressing the “**Save & Test**” button, and Grafana confirmed that the data source was working correctly. This connection allowed Grafana to pull metrics directly from Prometheus, making it possible to create visual dashboards.



Mihalache Raluca C00296281\_Lubuntu\_24/25      Enforce US Keyboard Layout | View Fullscreen | Send Ctrl+Alt+Delete

prometheus - Data sources - Connections - Grafana — Mozilla Firefox

Prometheus Time Series · prometheus - Data source · +

Not Secure 192.168.56.20:3000/connections/datasources/edit/aehppy...

Home > Connections > Data sources > prometheus

Cache level: Low

Incremental querying (beta)

Disable recording rules (beta)

Other

Custom query parameters: Example: max\_source\_resolution=5m&timeout=10s

HTTP method: POST

Use series endpoint

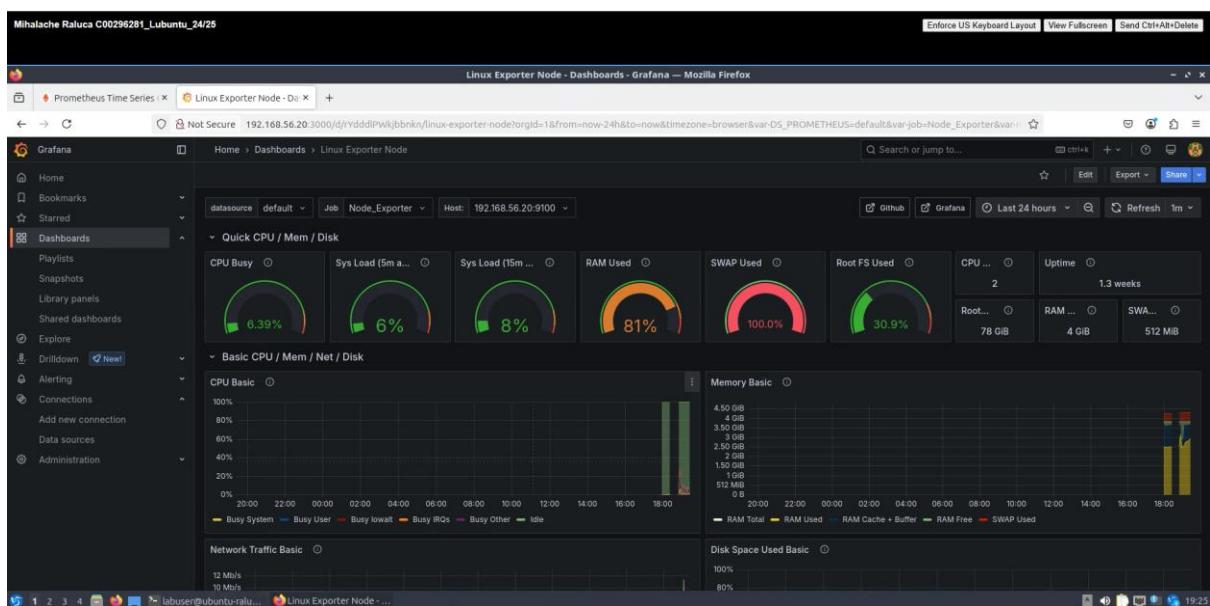
Exemplars

+ Add

✓ Successfully queried the Prometheus API.

Next, you can start to visualize data by [building a dashboard](#), or by querying data in the [Explore view](#).

Delete Save & test



To monitor Fail2ban with Prometheus, I first installed Fail2ban on my system. I then set up the Fail2ban Prometheus Exporter by cloning the official GitHub repository and building the binary using the **make build** command. After moving the binary to /usr/local/bin, I created a systemd service file to run the exporter as a background service. The exporter was

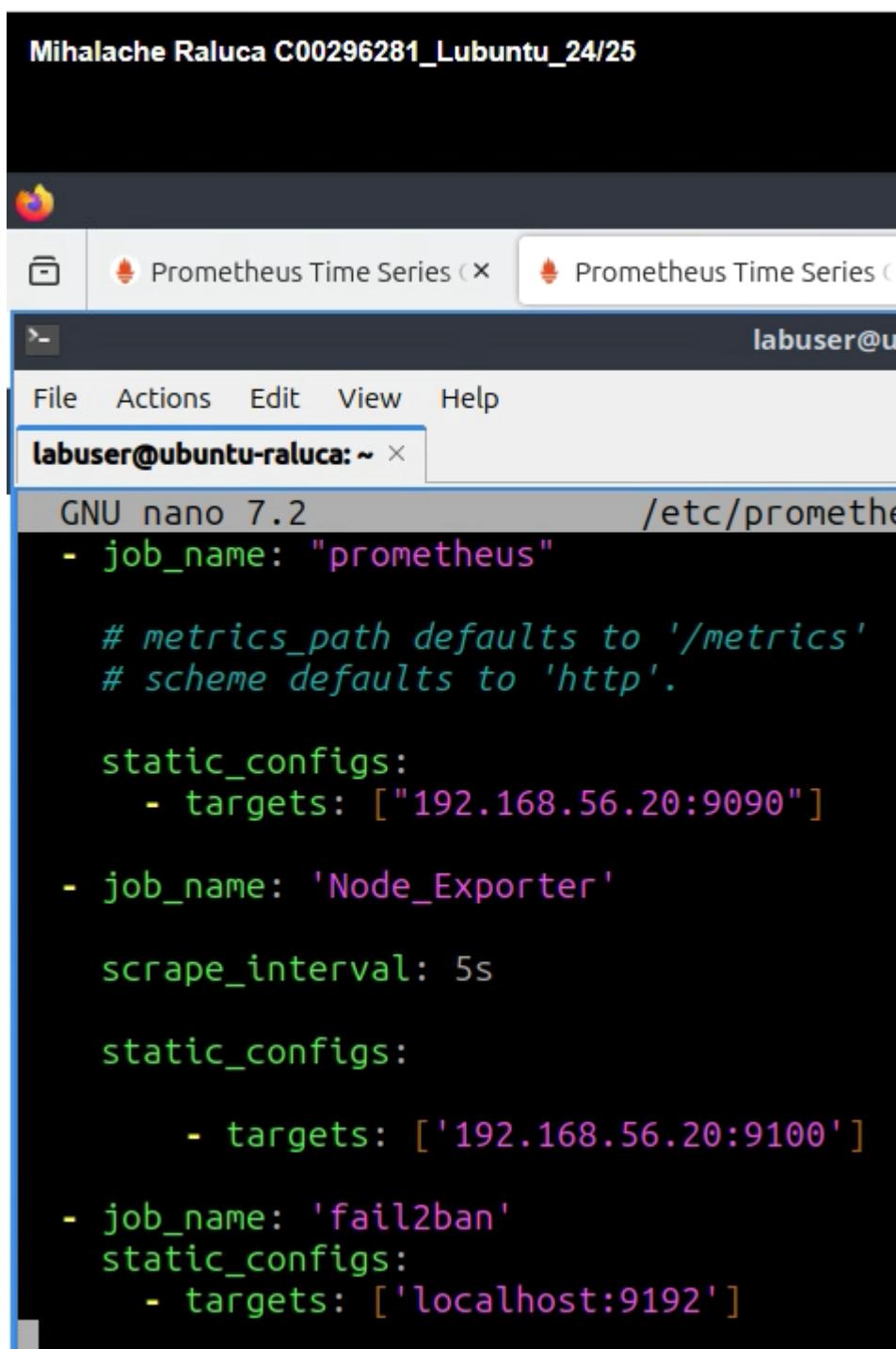
configured to listen to on port 9192 and expose metrics from Fail2ban. Once the service was enabled and running, I added a job to the prometheus.yml configuration file to scrape metrics from localhost:9192/metrics. After restarting Prometheus, the exporter successfully appeared in the Prometheus targets page.

The screenshot shows the Prometheus Targets page in a Mozilla Firefox browser. The URL is 192.168.56.20:9090/targets?search=. The page has a dark-themed header with the Prometheus logo and navigation links for Alerts, Graph, Status, and Help. Below the header, there's a search bar labeled "Filter by endpoint or labels".

The main content area is titled "Targets" and displays three groups of healthy endpoints:

- Node\_Exporter (1/1 up)**: Shows one endpoint at http://192.168.56.20:9100/metrics, labeled "UP" with "instance=192.168.56.20:9100" and "job=Node\_Exporter". Last scraped 359.000ms ago, duration 35.672ms.
- fail2ban (1/1 up)**: Shows one endpoint at http://localhost:9192/metrics, labeled "UP" with "instance=localhost:9192" and "job=fail2ban". Last scraped 7.897s ago, duration 4.869ms.
- prometheus (1/1 up)**: Shows one endpoint at http://192.168.56.20:9090/metrics, labeled "UP" with "instance=192.168.56.20:9090" and "job=prometheus". Last scraped 6.115s ago, duration 12.634ms.

Mihalache Raluca C00296281\_Lubuntu\_24/25



The screenshot shows a terminal window titled "labuser@ubuntu-raluca: ~" running the "GNU nano 7.2" editor. The file being edited is located at "/etc/prometheus/prometheus.yml". The content of the file is a YAML configuration for a Prometheus instance. It includes sections for job definitions, static configurations, and scrape intervals.

```
GNU nano 7.2 /etc/prometheus/prometheus.yml
- job_name: "prometheus"
  # metrics_path defaults to '/metrics'
  # scheme defaults to 'http'.
  static_configs:
    - targets: ["192.168.56.20:9090"]

- job_name: 'Node_Exporter'
  scrape_interval: 5s
  static_configs:
    - targets: ['192.168.56.20:9100']

- job_name: 'fail2ban'
  static_configs:
    - targets: ['localhost:9192']
```

Mihalache Raluca C00296281\_Lubuntu\_24/25      Enforce US Keyboard Layout      View Fullscreen

```

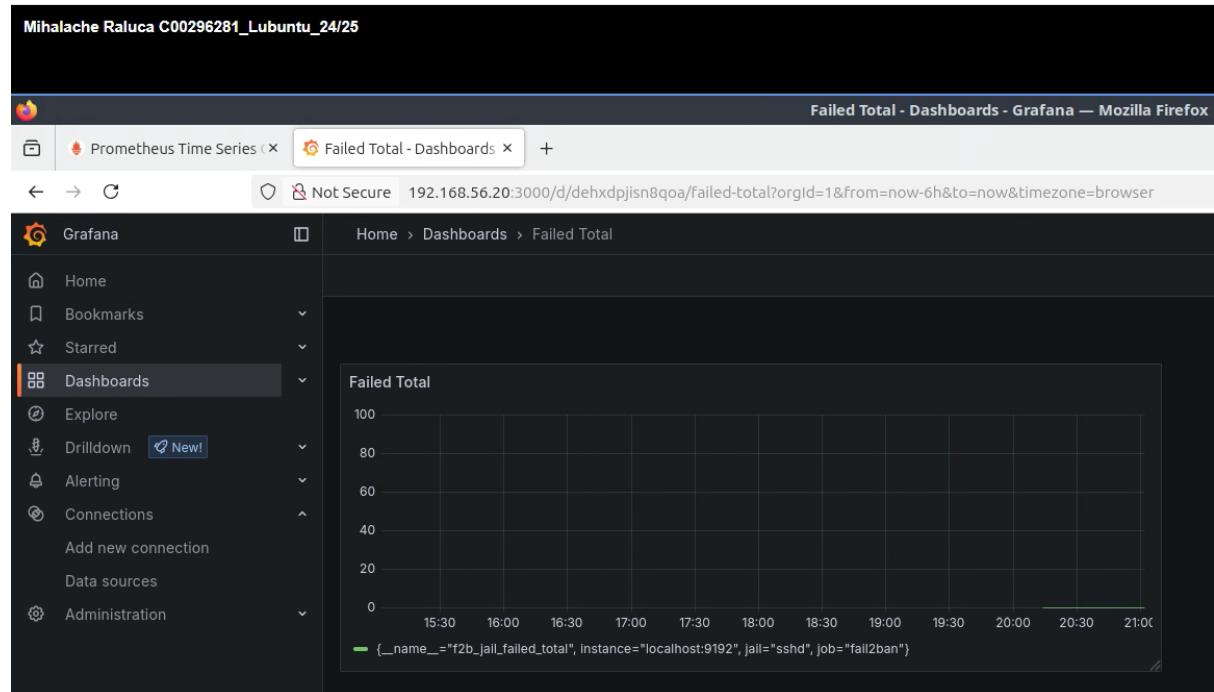
Prometheus Time Series Collection and Processing Server — Mozilla Firefox
Prometheus Time Series × Prometheus Time Series × + labuser@ubuntu-raluca: ~
File Actions Edit View Help labuser@ubuntu-raluca: ~ ×
GNU nano 7.2 /etc/systemd/system/fail2ban_exporter.service
[Unit]
Description=Fail2Ban Prometheus Exporter
After=network.target

[Service]
ExecStart=/usr/local/bin/fail2ban_exporter --web.listen-address=:9192"
User=root
Restart=always

[Install]
WantedBy=multi-user.target

```

To create the dashboard for Fail2ban in Grafana, I first made sure that Prometheus was successfully scraping metrics from the Fail2ban exporter. In Grafana, I created a new dashboard and added custom visualizations using Prometheus as the data source. I used the key metric f2b\_jail\_failed\_total to display the total number of failed login attempts.



To install Blackbox Exporter, I started by downloading the latest version of the Blackbox Exporter from GitHub using `wget`, then extracted the files and moved the binary to `/usr/local/bin`. I created a system user called `blackbox_exporter` and set up a configuration

file in etc/blackbox\_exporter/config.yml, which defines how different types of probes should behave. I then created a systemd service so the exporter would run in the background and start on boot. Finally, I configured Prometheus to scrape metrics from the Blackbox Exporter by adding a new scrape job in prometheus.yml and restarting the Prometheus service.

Node_Exporter (1/1 up)						
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error	
http://192.168.56.20:9100/metrics	UP	instance="192.168.56.20:9100" job="Node_Exporter"	1.924s ago	20.053ms		

blackbox (1/1 up)						
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error	
http://localhost:9115/probe	UP	instance="http://localhost:3000" job="blackbox" module="http_zxx" target="http://localhost:3000"	10.203s ago	132.448ms		

fail2ban (1/1 up)						
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error	
http://localhost:9192/metrics	UP	instance="localhost:9192" job="fail2ban"	4.463s ago	9.384ms		

prometheus (1/1 up)						
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error	
http://192.168.56.20:9090/metrics	UP	instance="192.168.56.20:9090" job="prometheus"	2.685s ago	12.994ms		

```
GNU nano 7.2 /etc/prometheus/prometheus.yml *
- job_name: "prometheus"
  static_configs:
    - targets: ["192.168.56.20:9090"]

- job_name: 'Node_Exporter'
  scrape_interval: 5s
  static_configs:
    - targets: ['192.168.56.20:9100']

- job_name: 'fail2ban'
  static_configs:
    - targets: ['localhost:9192']

- job_name: 'blackbox'
  metrics_path: /probe
  params:
    module: [http_2xx]
  static_configs:
    - targets:
        - http://localhost:3000
  relabel_configs:
    - source_labels: [__address__]
      target_label: __param_target
    - source_labels: [__param_target]
      target_label: instance
    - target_label: __address__
      replacement: localhost:9115 # Blackbox Exporter address
```

## Mihalache Raluca C00296281\_Lubuntu\_24/25

```
labuser@ubuntu-raluca: ~/blackbox_exporter-0.24.0.linux-amd64
```

```
GNU nano 7.2 /etc/blackbox_exporter.conf
```

```
modules:
```

```
  http_2xx:
```

```
    prober: http
```

```
    timeout: 5s
```

```
  http:
```

```
    method: GET
```

```
    fail_if_ssl: false
```

```
  tcp_connect:
```

```
    prober: tcp
```

```
    timeout: 10s
```

```
  tcp:
```

```
    preferred_ip_protocol: "ip4"
```

## Mihalache Raluca C00296281\_Lubuntu\_24/25

Enforce US Keyboard Layout View Fullscreen Send Ctrl+Alt+Delete

```
labuser@ubuntu-raluca: ~/blackbox_exporter-0.24.0.linux-amd64
```

```
GNU nano 7.2 /etc/systemd/system/blackbox_exporter.service
```

```
[Unit]
```

```
Description=Prometheus Blackbox Exporter
```

```
After=network.target
```

```
[Service]
```

```
User=blackbox_exporter
```

```
Group=blackbox_exporter
```

```
Type=simple
```

```
ExecStart=/usr/local/bin/blackbox_exporter --config.file=/etc/blackbox_exporter/config.yaml
```

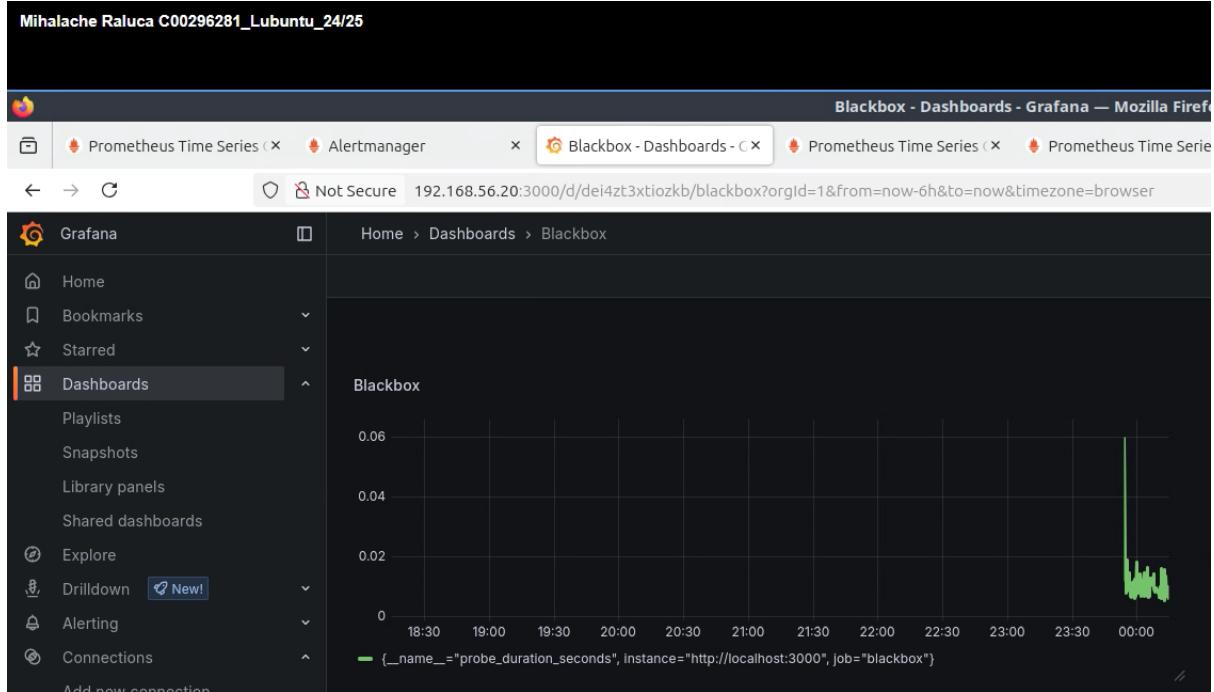
```
Restart=always
```

```
[Install]
```

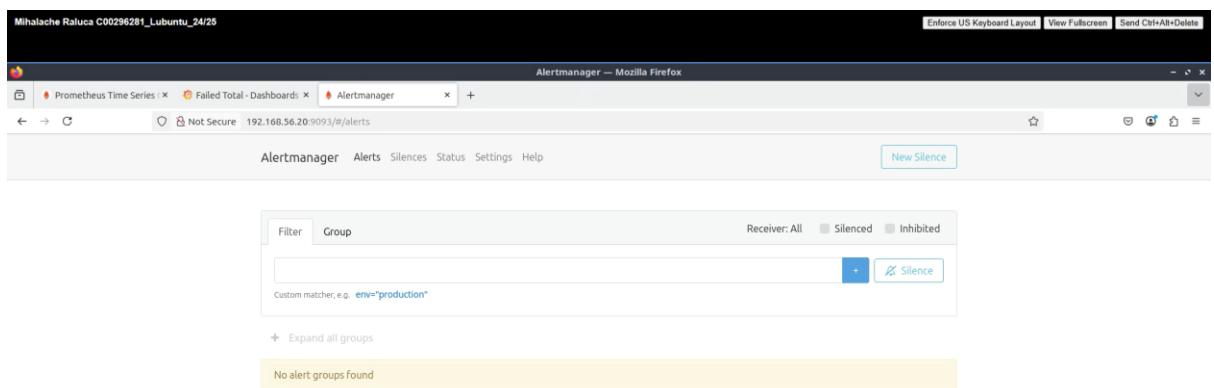
```
WantedBy=multi-user.target
```

For my Blackbox Exporter dashboard in Grafana, I chose to monitor the target <http://192.168.56.20:3000>, which is the IP address and port local of my Grafana instance. I

used the metric **probe\_duration\_seconds**, to track how long it takes for the Blackbox Exporter to complete a HTTP probe.



To enable alert notifications for potential security events, I installed Prometheus Alertmanager on my Ubuntu server. I began by downloading the latest Alertmanager release, extracting the files, and moving the binaries to the system path. I then created a configuration file **alertmanager.yml** that defines the routing rules and default receivers. A system service was also created to ensure Alertmanager runs in the background and starts on boot. After configuring the service and creating the necessary system directories and user permissions, I started and enabled Alertmanager. I confirmed that it was running by accessing the Alertmanager web interface on port 9093.



Alertmanager — Mozilla Firefox

Prometheus Time Series Failed Total - Dashboards Alertmanager

labuser@ubuntu-raluca: ~/alertmanager-0.27.0.linux-amd64

File Actions View Help

labuser@ubuntu-raluca: ~/alertmanager-0.27.0.linux-amd64

```
GNU nano 7.2          /etc/systemd/system/alertmanager.service

[Unit]
Description=Prometheus Alertmanager
Wants=network-online.target
After=network-online.target

[Service]
User=alertmanager
Group=alertmanager
Type=simple
ExecStart=/usr/local/bin/alertmanager --config.file=/etc/alertmanager/alertmanager.yaml
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

[ Read 14 lines ]

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location  
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line

The screenshot shows a terminal window titled "Alertmanager — Mozilla Firefox" running on a Lubuntu system. The terminal is displaying the configuration file for an Alertmanager instance, specifically `/etc/alertmanager/alertmanager.yml`. The configuration includes sections for global settings, routing, receivers, and inhibit rules.

```
GNU nano 7.2 /etc/alertmanager/alertmanager.yml
global:
  resolve_timeout: 5m

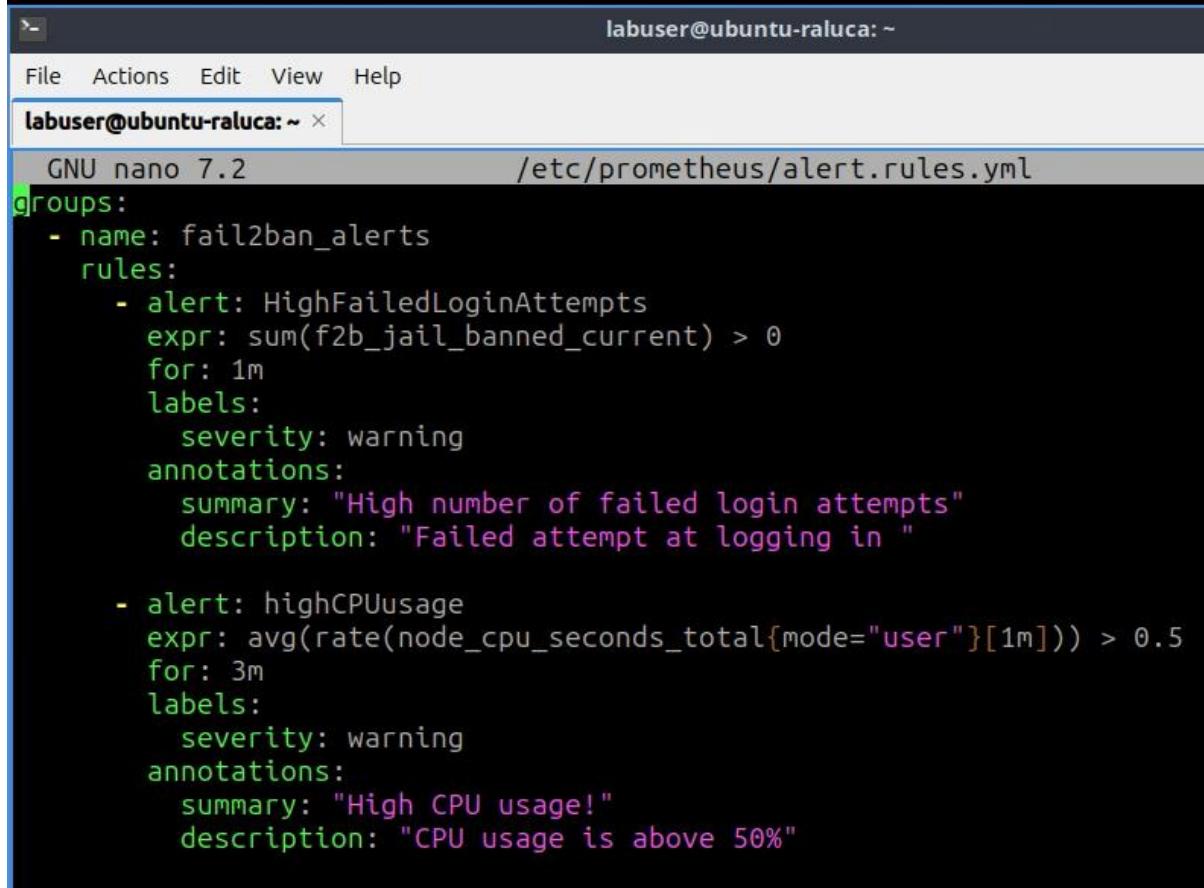
route:
  receiver: 'default'

receivers:
- name: 'default'

inhibit_rules:
- source_match:
    severity: 'critical'
  target_match:
    severity: 'warning'
    equal: ['alertname', 'cluster', 'service']
```

To monitor the system more effectively, I added customer alerts in Prometheus by editing the alert.rules.yml file. I created two alert rules under a group called fail2ban\_alerts. The first rule, HighFailedLoginAttempts checks if there are any IPs currently banned by Fail2ban due too many failed SSH login attempts. The second rule, highCPUUsage checks if the CPU usage is above 50% over a 3 minute period and triggers a warning alert if true.

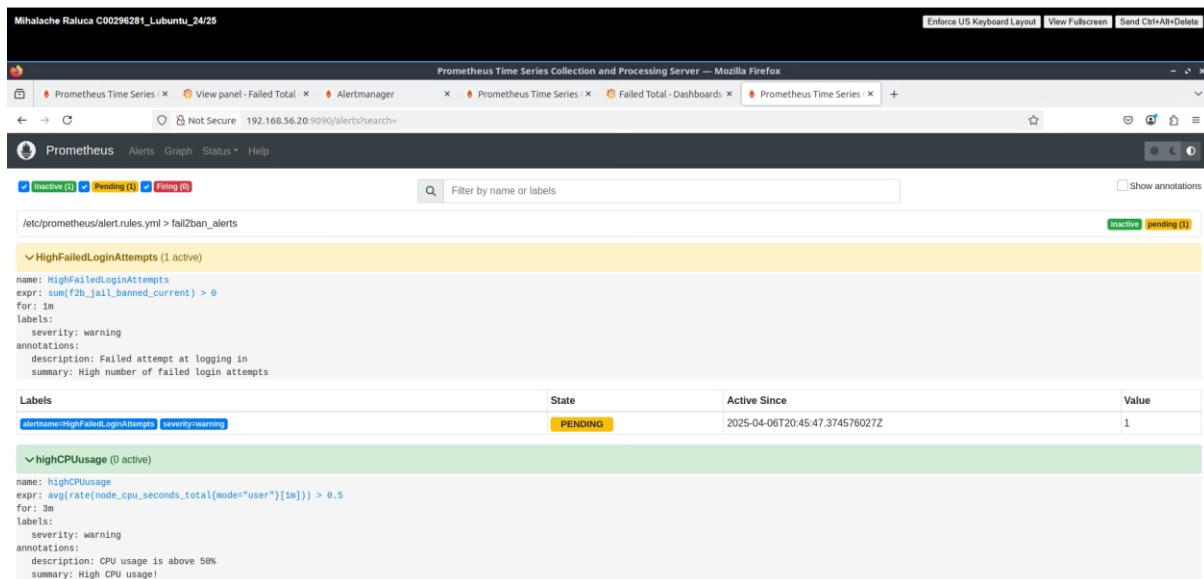
## Mihalache Raluca C00296281\_Lubuntu\_24/25



```
GNU nano 7.2          /etc/prometheus/alert.rules.yml

groups:
- name: fail2ban_alerts
  rules:
    - alert: HighFailedLoginAttempts
      expr: sum(f2b_jail_banned_current) > 0
      for: 1m
      labels:
        severity: warning
      annotations:
        summary: "High number of failed login attempts"
        description: "Failed attempt at logging in"

    - alert: highCPUUsage
      expr: avg(rate(node_cpu_seconds_total{mode="user"}[1m])) > 0.5
      for: 3m
      labels:
        severity: warning
      annotations:
        summary: "High CPU usage!"
        description: "CPU usage is above 50%"
```



Prometheus Time Series Collection and Processing Server — Mozilla Firefox

192.168.56.20:9090/alerts?search=

Prometheus Alerts Graph Status Help

Filter by name or labels

/etc/prometheus/alert.rules.yml > fail2ban\_alerts

HighFailedLoginAttempts (1 active)

```
name: HighFailedLoginAttempts
expr: sum(f2b_jail_banned_current) > 0
for: 1m
labels:
  severity: warning
annotations:
  description: Failed attempt at logging in
  summary: High number of failed login attempts
```

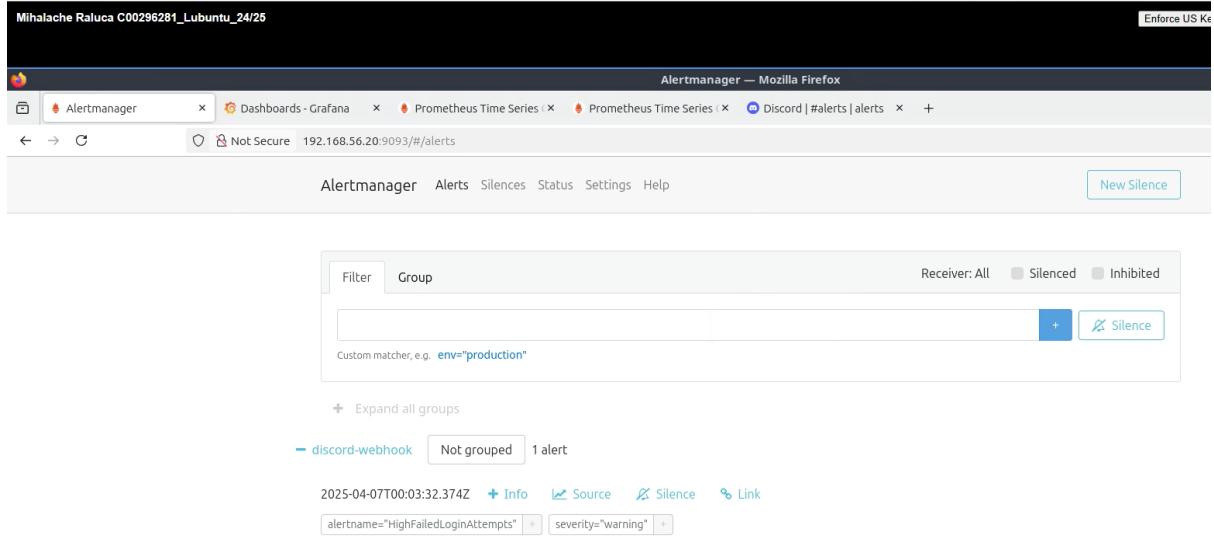
Labels	State	Active Since	Value
alarmname=HighFailedLoginAttempts severity=warning	PENDING	2025-04-06T20:45:47.374576027Z	1

highCPUUsage (0 active)

```
name: highCPUUsage
expr: avg(rate(node_cpu_seconds_total{mode="user"}[1m])) > 0.5
for: 3m
labels:
  severity: warning
annotations:
  description: CPU usage is above 50%
  summary: High CPU usage!
```

To get Alertmanager to send alerts to discord , I first created a new text channel on my discord server and set up a new webhook for that channel. Then I edited /etc/alertmanager/alertmanager.yml to include a new receiver called discword-webhook, where I specified the Discord webhook URL under webhook\_configs. I also set the route to

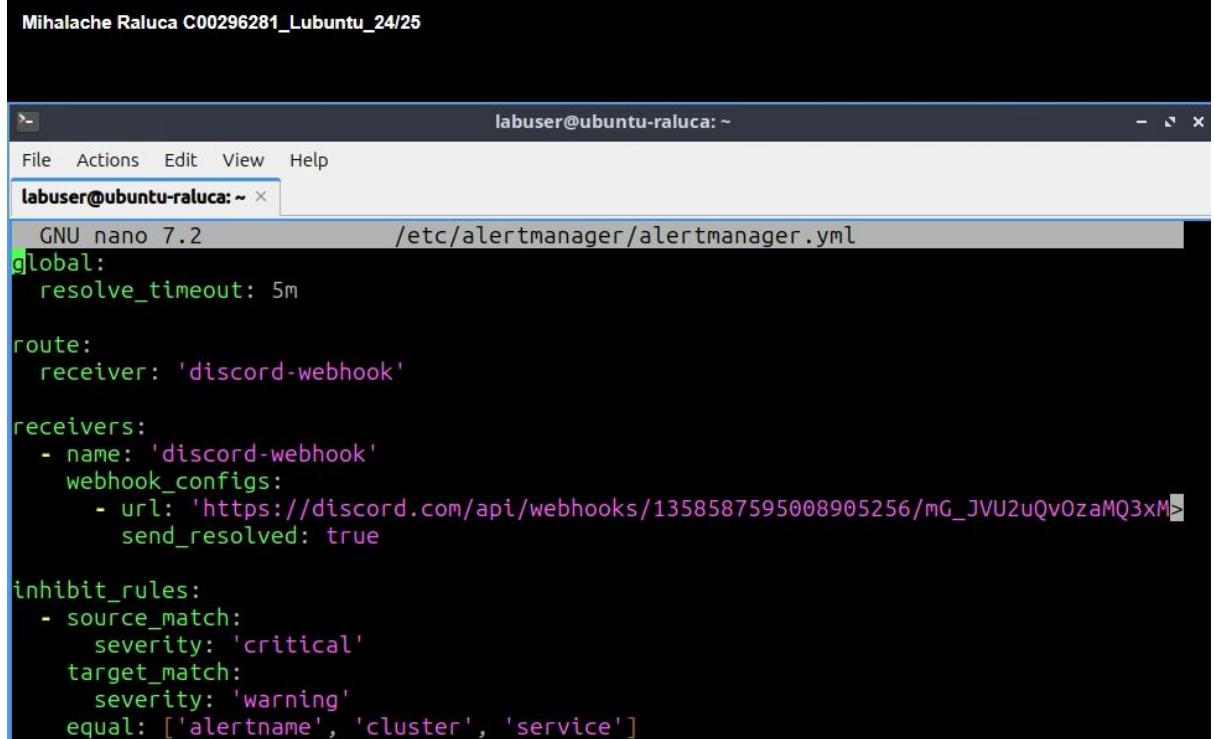
send all alerts to this new receiver. After saving the file, I restarted Alertmanager to apply the changes.



The screenshot shows the Alertmanager interface in Mozilla Firefox. The URL is 192.168.56.20:9093/#/alerts. The main area displays a single alert titled "HighFailedLoginAttempts". The alert details are as follows:

- Timestamp: 2025-04-07T00:03:32.374Z
- Severity: warning
- Source: Info
- Silence: Silence
- Link: Link
- Alertname: "HighFailedLoginAttempts"

Below the alert, there is a "New Silence" button.

The screenshot shows a terminal window with the command "nano /etc/alertmanager/alertmanager.yml" running. The configuration file contains the following YAML code:

```
global:
  resolve_timeout: 5m

route:
  receiver: 'discord-webhook'

receivers:
  - name: 'discord-webhook'
    webhook_configs:
      - url: 'https://discord.com/api/webhooks/1358587595008905256/mG_JVU2uQvOzaMQ3xM'
        send_resolved: true

inhibit_rules:
  - source_match:
      severity: 'critical'
    target_match:
      severity: 'warning'
    equal: ['alertname', 'cluster', 'service']
```

To test SSH protection with Fail2ban, I used my Rocky Linux server to simulate repeated failed login attempts on my Ubuntu server. I ran the SSH command from the Rocky Linux server using an invalid username e.g. `ssh laboser@192.168.56.20`. After several failed attempts, I checked the status using `sudo fail2ban-client status sshd`, which confirmed that the IP address of the Rocky Linux server had been banned. This verified that Fail2ban's detection and prevention were successfully working. I also set ban rules on repeated failed login attempts, by configuring the SSH jail in the Fail2ban jail.local file. I defined specific prevention criteria by setting `maxretry` to 5, `findtime` to 10 minutes, and `bantime` to 20

minutes. This means that if a user fails to log in via SSH five times within a 10 minute window, their IP addresses will be automatically banned for 20 minutes.

```
labuser@ubuntu-raluca:~$ sudo fail2ban-client status sshd
[sudo] password for labuser:
Status for the jail: sshd
|- Filter
| |- Currently failed: 1
| |- Total failed:    13
| ` Journal matches: _SYSTEMD_UNIT=sshd.service + _COMM=sshd
`- Actions
  |- Currently banned: 0
  |- Total banned:     1
  `- Banned IP list:
[labuser@ubuntu-raluca:~$
```

Mihalache Raluca C00296281\_RockyLinux\_24/25

```
Rocky Linux 8.10 (Green Obsidian)
Kernel 4.18.0-553.e18_10.x86_64 on an x86_64

Activate the web console with: systemctl enable --now cockpit.socket

rocky-raluca login: labuser
Password:
Last login: Mon Mar 24 05:01:39 on tty1
[labuser@rocky-raluca ~]$ sudo apt update -y
[sudo] password for labuser:
sudo: apt: command not found
[labuser@rocky-raluca ~]$ apt update -y
-bash: apt: command not found
[labuser@rocky-raluca ~]
[labuser@rocky-raluca ~]$ ssh laboser@192.168.56.20
ssh: connect to host 192.168.56.20 port 22: Connection refused
[labuser@rocky-raluca ~]$ ssh laboser@192.168.56.20
The authenticity of host '192.168.56.20 (192.168.56.20)' can't be established.
ECDSA key fingerprint is SHA256:Jtfg0JhZDgIonWQNxh7mZW70lpDe1Lyht7hq9NR+k4.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.56.20' (ECDSA) to the list of known hosts.
laboser@192.168.56.20's password:
Permission denied, please try again.
laboser@192.168.56.20's password:
Permission denied, please try again.
laboser@192.168.56.20's password:
laboser@192.168.56.20: Permission denied (publickey,password).
[labuser@rocky-raluca ~]$ _
```

Mihalache Raluca C00296281\_Lubuntu\_24/25

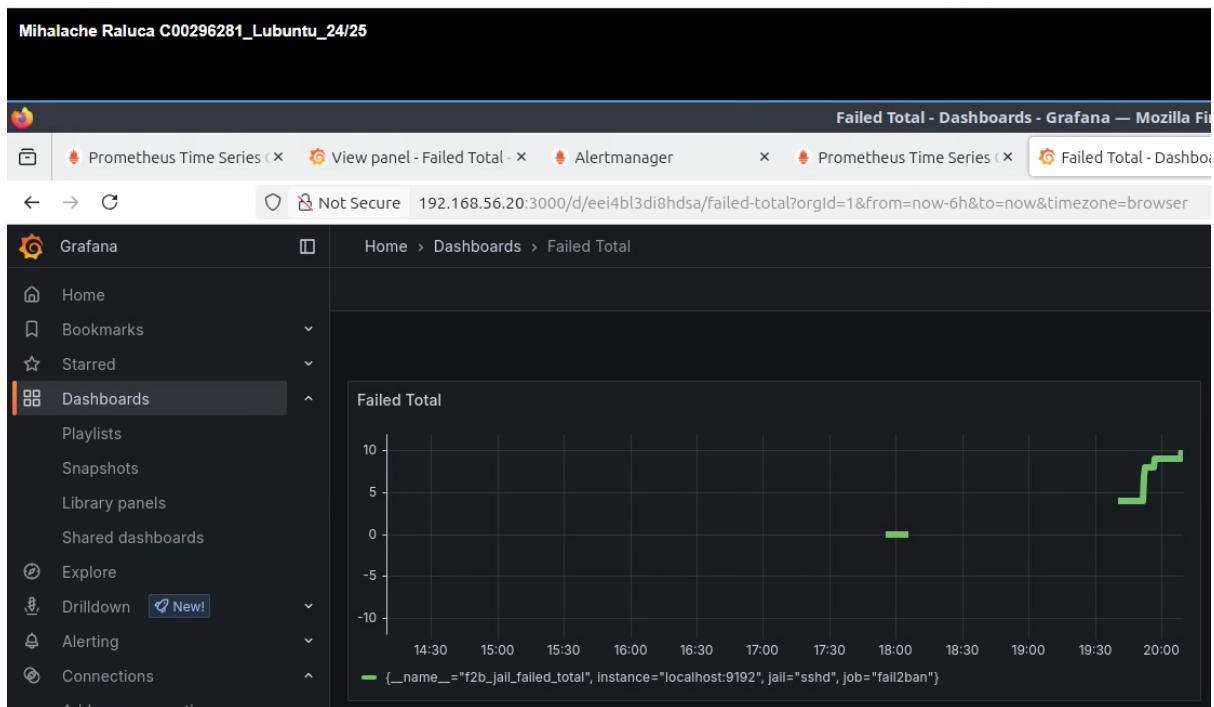
```
File Actions Edit View Help
labuser@ubuntu-raluca: ~ ×
GNU nano 7.2
#
# ignorecommand = /path/to/command
ignorecommand =

# "bantime" is the number of seconds
bantime = 20m

# A host is banned if it has generated
# seconds.
findtime = 10m

# "maxretry" is the number of failed logins before
# the account is banned.
maxretry = 5
```

I also checked the Grafana dashboard for Fail2ban, where I was able to visualize the metrics.



## Resources

[Install Prometheus and Grafana on Ubuntu 24.04 LTS](#)

[Step-by-step guide to setting up Prometheus Alertmanager with Slack, PagerDuty, and Gmail | Grafana Labs](#)

[How to Install Prometheus Alertmanager on Ubuntu Server : Step-by-step guide - Khamlou Digital](#)  
[Installing Prometheus Alertmanager](#)

[How to Install Prometheus Blackbox Exporter on Ubuntu: A Step-by-Step Guide - Khamlou Digital](#)  
[PromLabs | Blog - Sending Prometheus Alerts to Discord with Alertmanager v0.25.0](#)