**Monitoring Demo**

**Agenda:**

# Install Prometheus/ Node Exporter

# Pointing to a domain name

# Reverse proxy with NGINX

# Add SSL to Prometheus Reverse Proxy

# Add basic authentication to the Prometheus User Interface

# Scrape Target Basics

# Add rules

# Install Prometheus AlertManager

# Install Grafana

**Prerequisites:**

All commands were tested on a **Ubuntu 22.04 LTS**. If you chose to use another distribution, you might need to check appropriate commands to get the same result.

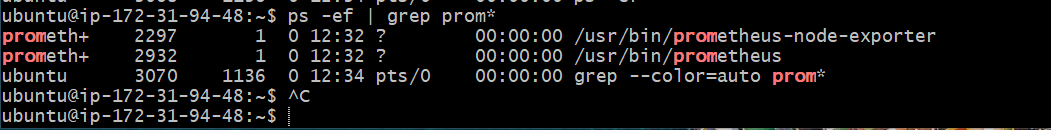
1. Have access to an unrestricted Ubuntu server. I got mine from <https://www.digitalocean.com> but you can use other providers as long as they don’t have extra firewall configuration that may interfere with this demo
2. You will need to have root access
3. To be able to do the DNS configuration part, you also need to have a domain purchased in advance.
4. **Install Prometheus/ Node Exporter through the package manager from Ubuntu**

First, we are gonna update packages and install prometheus:

| sudo apt update  sudo apt install prometheus |
| --- |

This install will set up a Prometheus process and also a Node Exporter process.

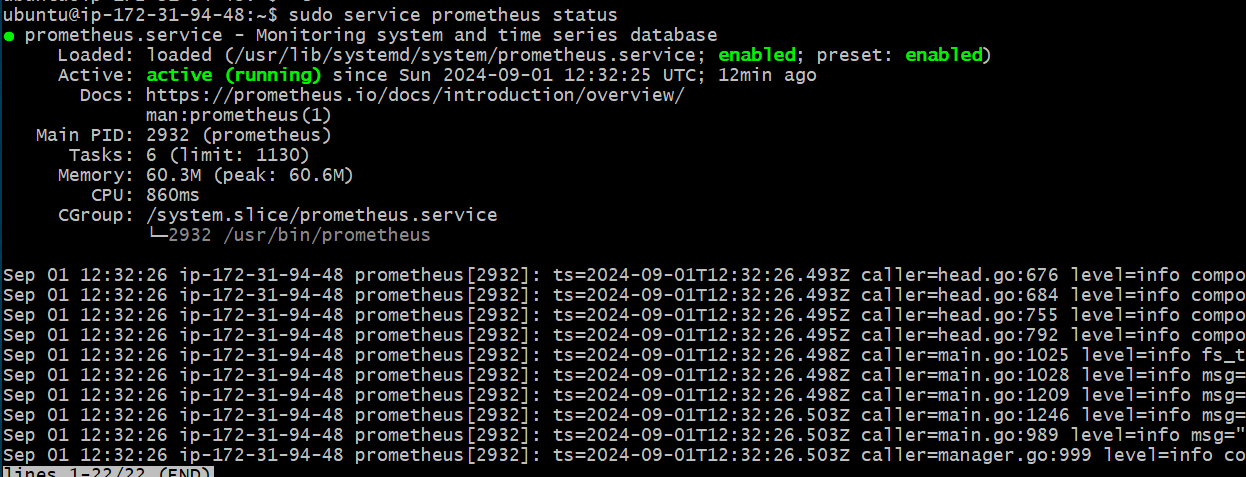
| ubuntu@ip-172-31-94-48:~$ ps -ef | grep prom\*  prometh+ 2297 1 0 12:32 ? 00:00:00 /usr/bin/prometheus-node-exporter  prometh+ 2932 1 0 12:32 ? 00:00:00 /usr/bin/prometheus |
| --- |



NOTE: Prometheus will listen on port 9090 and node exporter on port 9100.

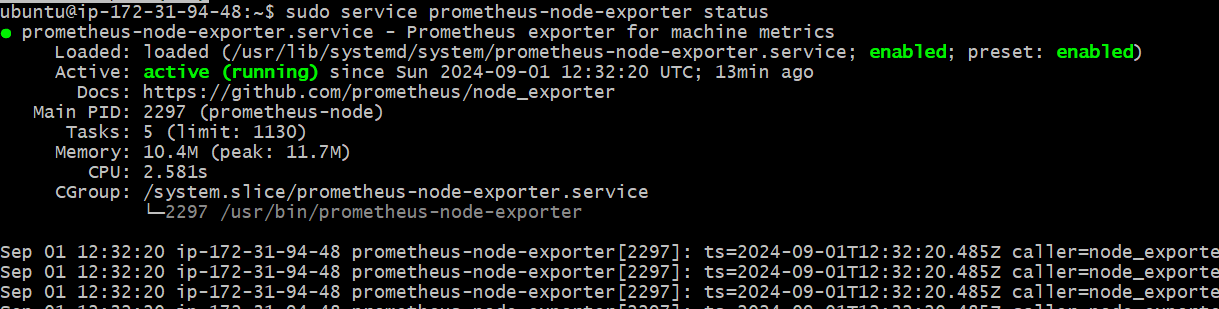
**To check those services are actually running:**

| sudo service prometheus status |
| --- |



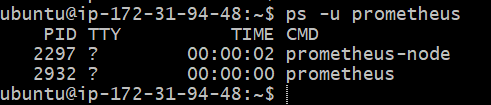
Similar to get node exporter status:

| sudo service prometheus-node-exporter status |
| --- |

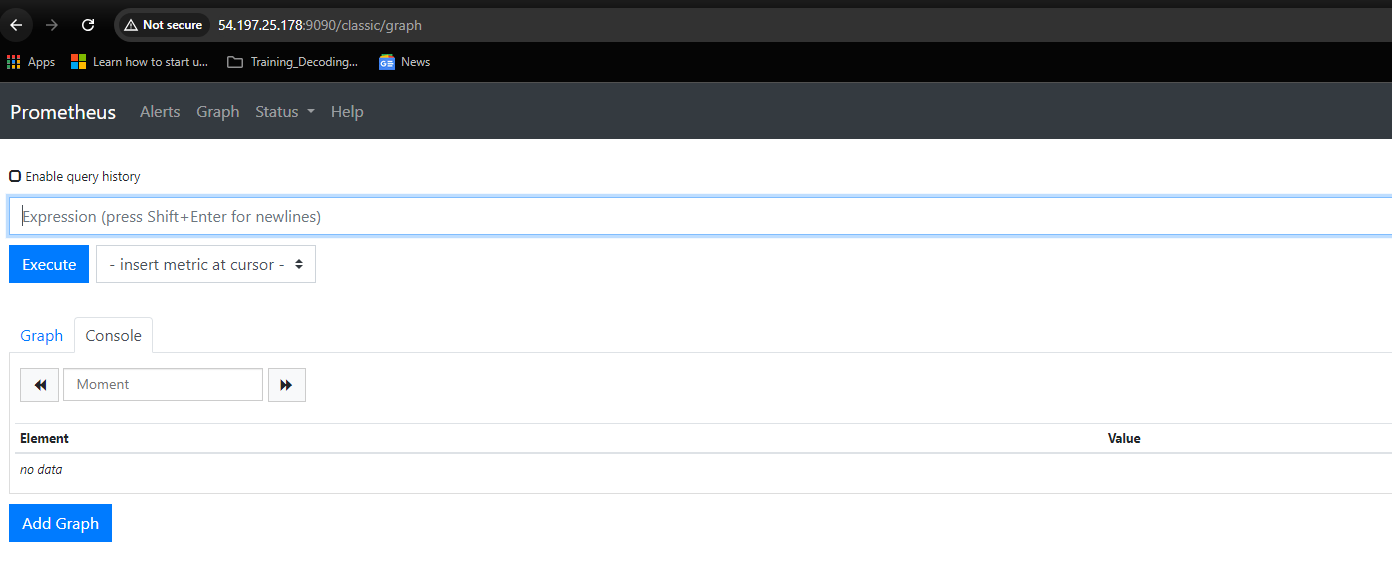


We also get a user called prometheus, which is running 2 processes:

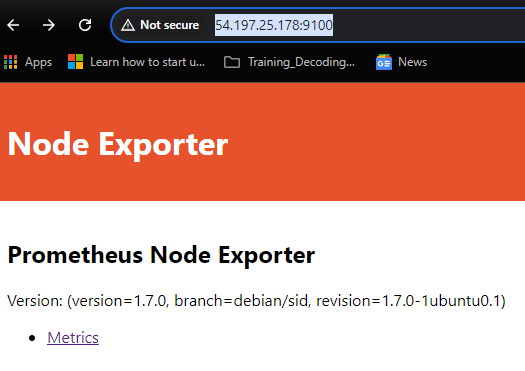
| ps -u prometheus |
| --- |



If we don't have any FW to configure, we can already use the machine public IP + the prometheus port to get a first look at our prometheus instance:



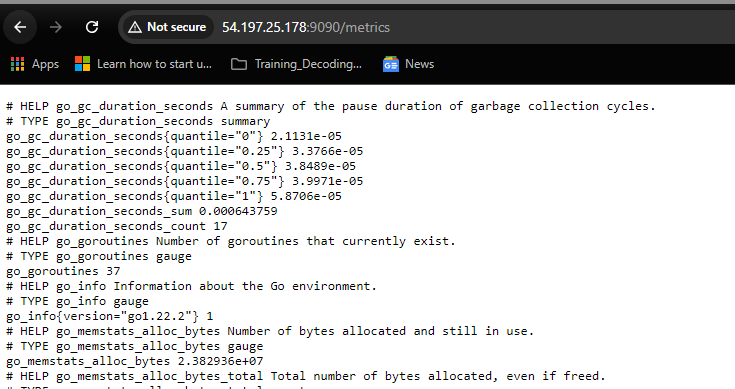
Or at our Node Exporter instance: <http://54.197.25.178:9100/>



Prometheus will query this 9100/metrics endpoint at intervals and we will be able to view these metrics in prometheus as time series data;

Prometheus also has a metrics endpoints, and will be reading data from both of them:

<http://54.197.25.178:9090/metrics>



At this moment Prometheus is accessible from the internet, we can do a little tweaks to lock it down:

1. give it an SSL certificate
2. a domain name
3. setup basic authentication so you can’t access it without an username & password;

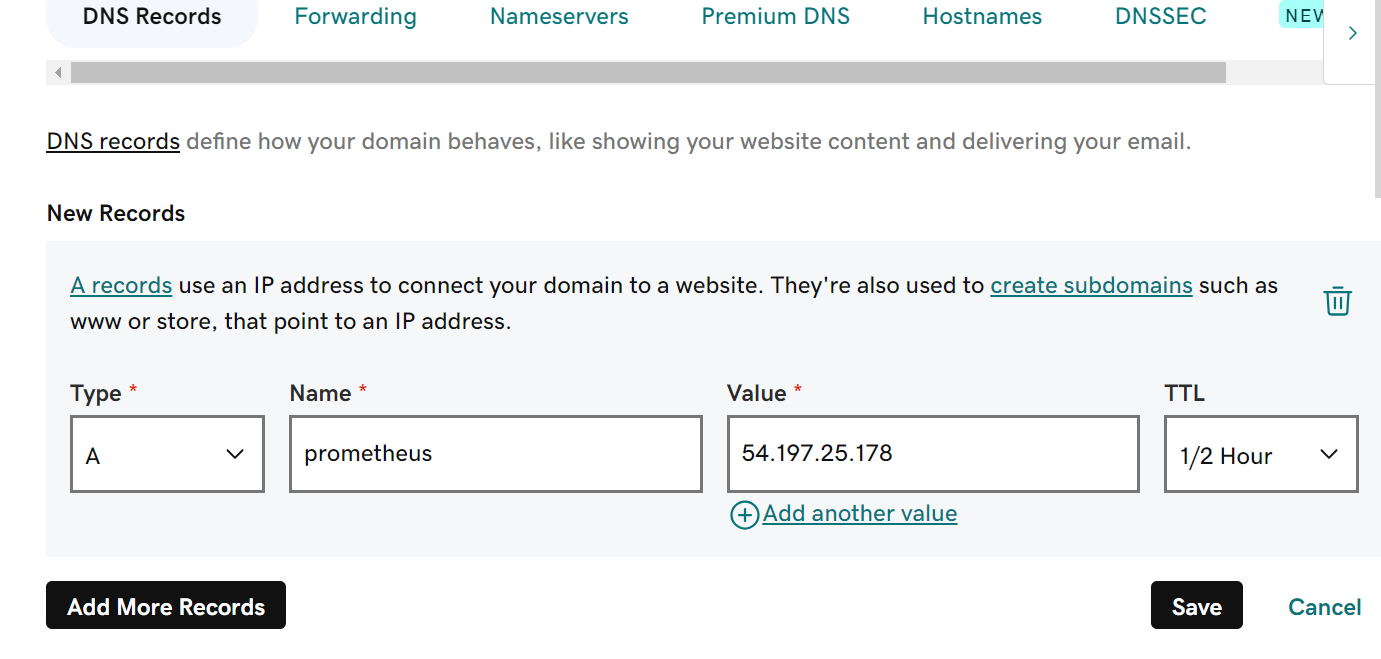
# **2. Pointing to a domain name**

**Current config:**  We can access Prometheus instance using our instance IP Example: <http://54.197.25.178:9090/>

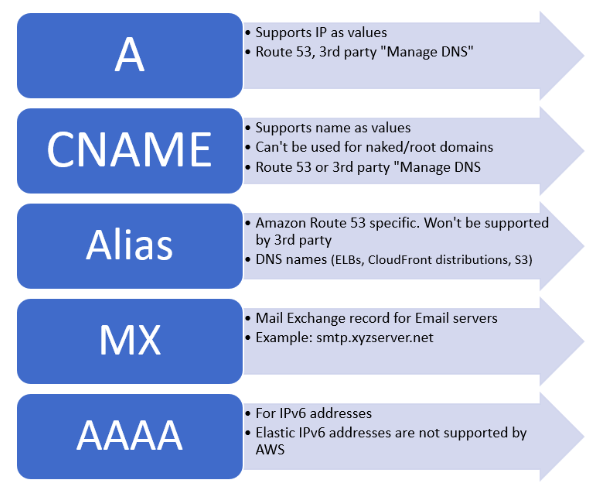
**Target config:** access it using DNS. Example: <https://prometheus.domain> name

**Prerequisites: have a domain purchased;**

Go to your domain purchased prom your preferred provider and Add a A tag domain:

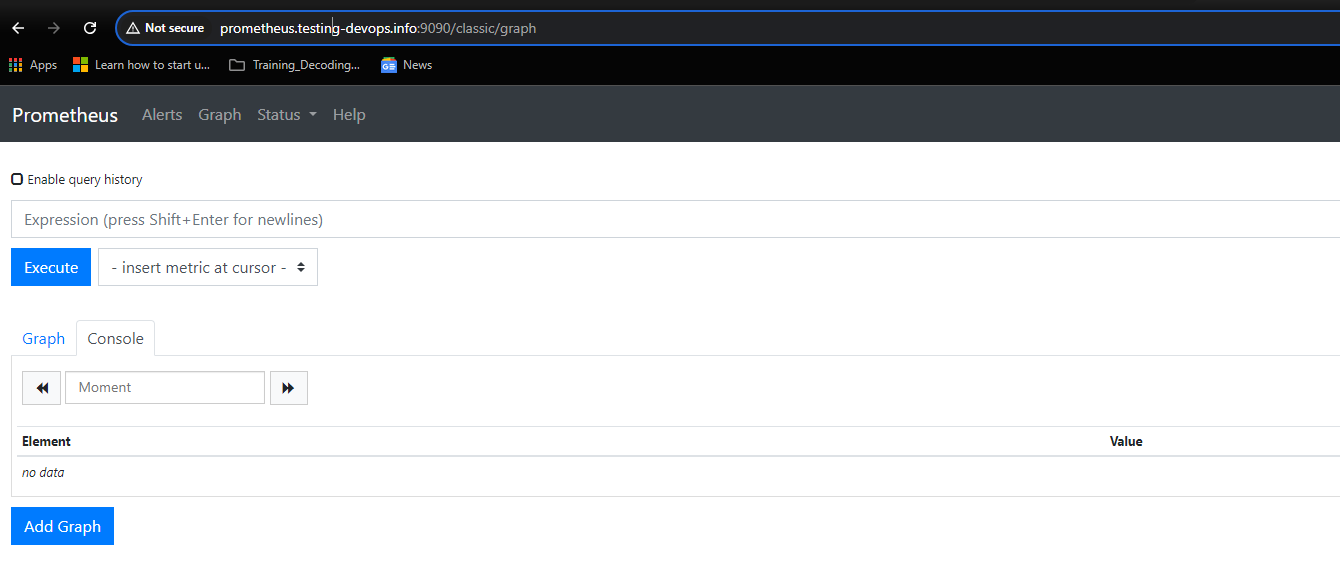


**Definitions:**



**Note:** Most DNS updates take effect within an hour, but could take up to 48 hours to update globally.

We need to check: [http://prometheus.testing-devops.info:9090/](https://prometheus.testing-devops.info:9090/) after a while;

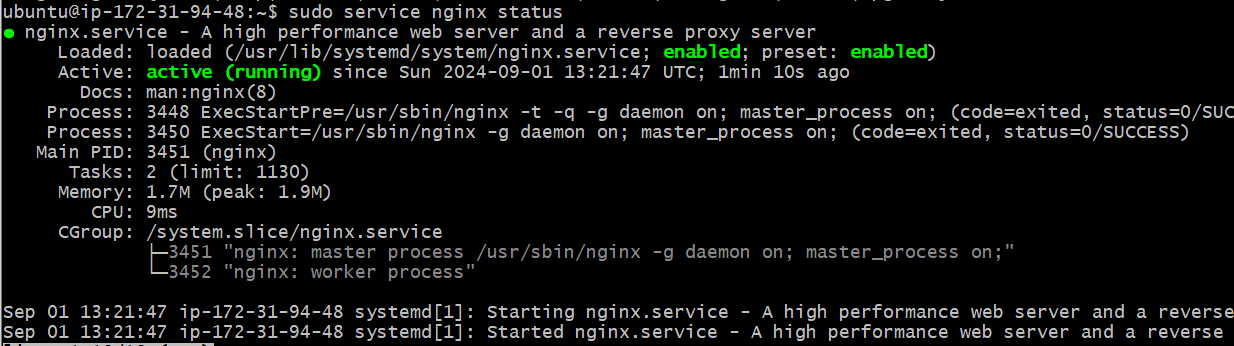


**Next step is set it up under reverse proxy so we don't have to use the port any longer and give it an SSL connection;**

# **3. Reverse proxy with NGINX**

First, install nginx and check its status:

| sudo apt install nginx  sudo service nginx status |
| --- |



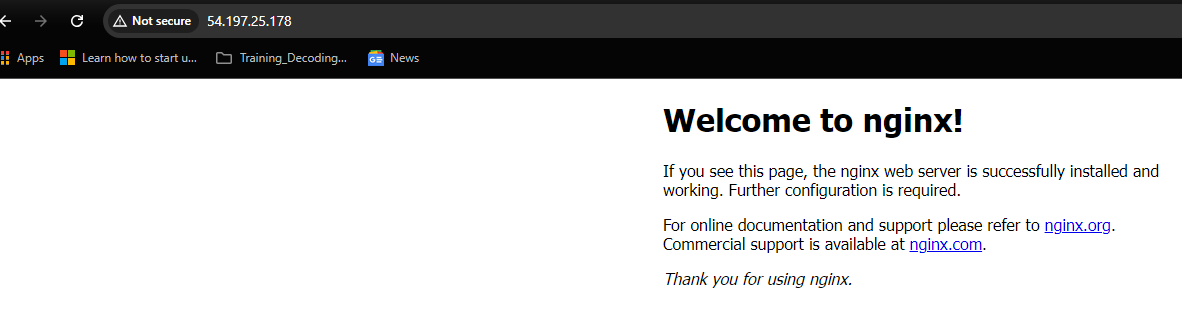
cd to NGINX folder:

| cd /etc/nginx/sites-enabled |
| --- |

In here we will get the default page we get when we install nginx:

lrwxrwxrwx 1 root root 34 Sep 1 13:21 **default -> /etc/nginx/sites-available/default**

First screen available based on machine IP:



**Note:** it listens on port 80 by default so we don’t have to type it;

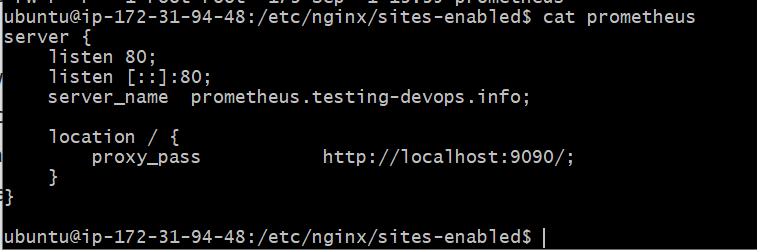
We will create a new config. Switch to root with sudo -i in case you were not from beginning:

| sudo vi prometheus |
| --- |

**and then insert our Prometheus server name:**

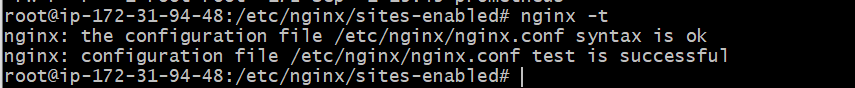
| server {  listen 80;  listen [::]:80;  server\_name YOUR-DOMAIN-NAME; => prometheus.testing-devops.info;  location / {  proxy\_pass http://localhost:9090/;  }  } |
| --- |

So now that we added server name, it is going to forward that default location into 9090, the default port that prometheus listens to;



We can also test the new configuration has no errors:

| nginx -t |
| --- |



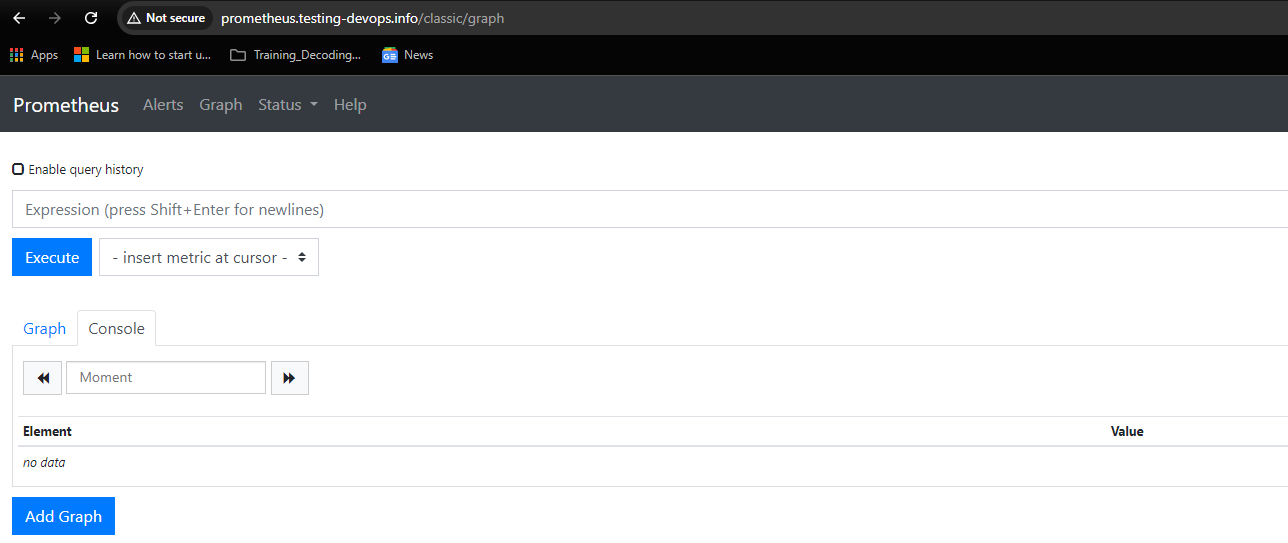
Next, restart nginx to load recent updates:

| root@ip-172-31-94-48:/etc/nginx/sites-enabled# **sudo service nginx restart**  root@ip-172-31-94-48:/etc/nginx/sites-enabled# **sudo service nginx status** |
| --- |



Now if we check the url in the browser, it will direct us to prometheus, with no need to use the port:

<http://prometheus.testing-devops.info/classic/graph>

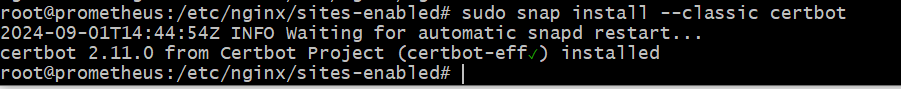


Next we will also set up SSL, as at the moment our connection is not secure:

# **Add SSL to Prometheus Reverse Proxy**

1 way to get a certificate from: <https://certbot.eff.org/instructions?ws=nginx&os=snap>

| sudo snap install --classic certbot |
| --- |

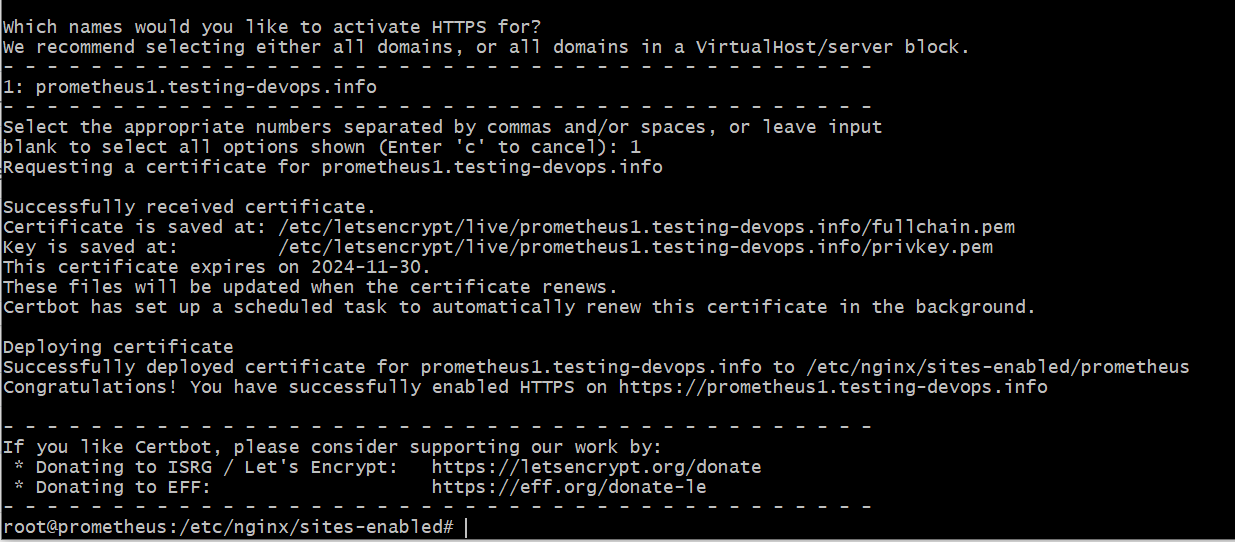


| sudo certbot --nginx |
| --- |

Answer to provided questions:

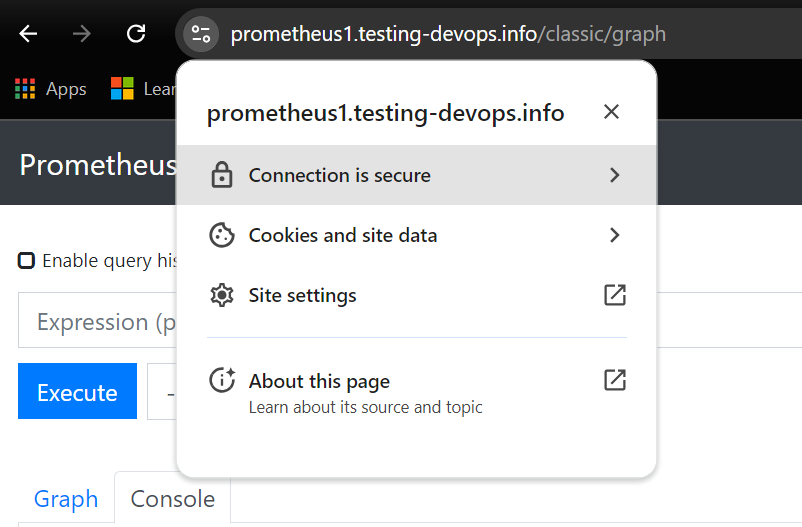
* Enter email address: enter your email address
* Accept terms & conditions: Y
* Campaigns: N
* Select the domain: option 1

It will scan the nginx config and find the website:



After installing the SSL cert, the https page will work:

https://prometheus1.testing-devops.info/classic/graph



Another thing, if we type in http, now it will FW it to https automatically;

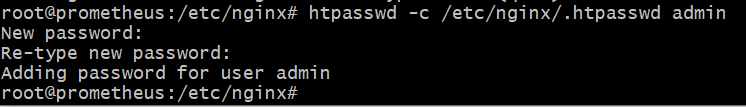
Also, if we check out the prometheus config file from sites-enabled, it will have more information added to support SSL configuration, extra config added automatically by certbot;

1. **Add basic authentication to the Prometheus User Interface:**

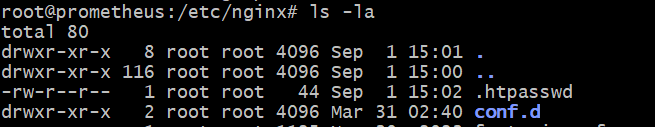
| cd /etc/nginx  sudo apt install apache2-utils |
| --- |

To create a password file for an user called admin:

| htpasswd -c /etc/nginx/.htpasswd admin |
| --- |

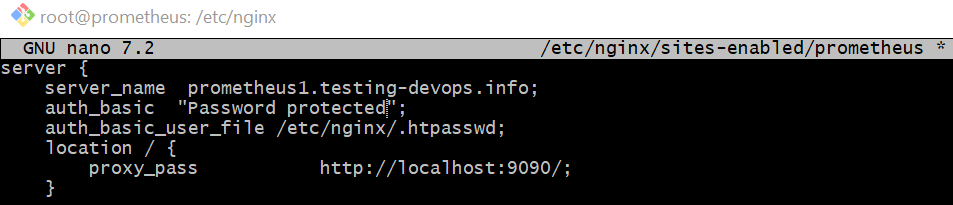


New file password created:

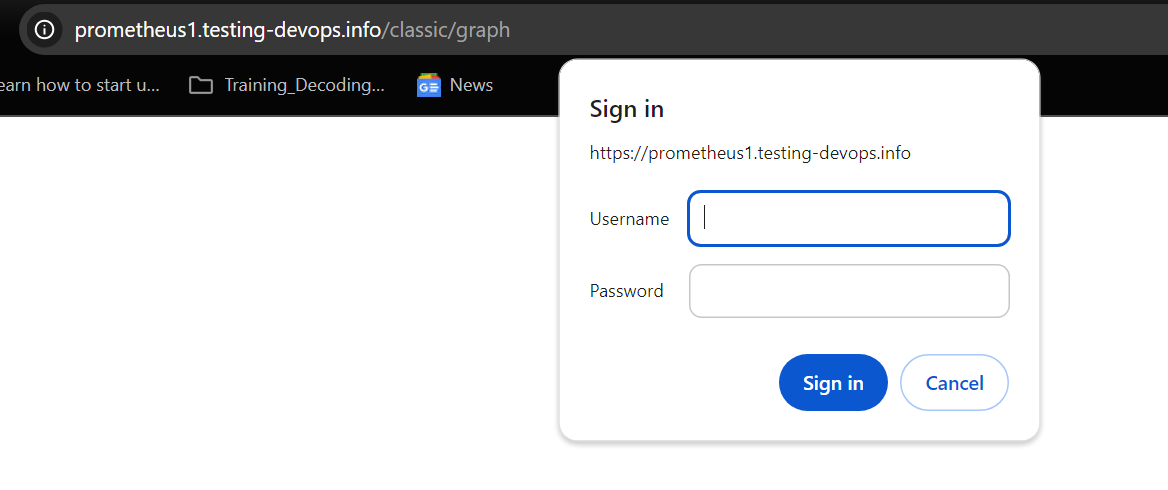


Now, in the prometheus file that we have under nginx sites-enabled path, we will add a pointer to the user file.

| auth\_basic "Password Protected";  auth\_basic\_user\_file /etc/nginx/.htpasswd; |
| --- |



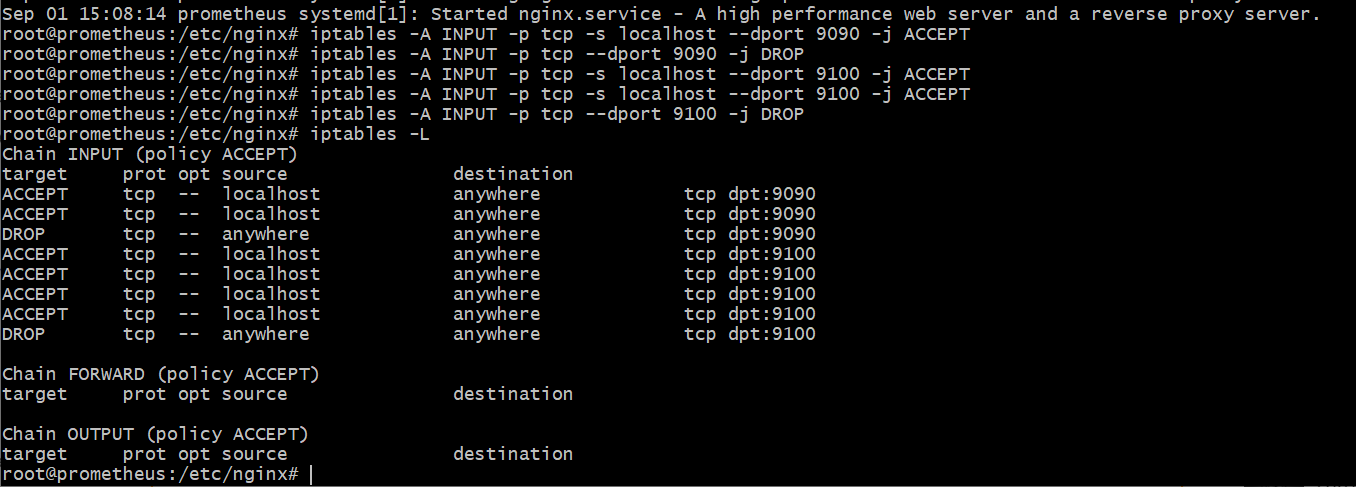
Restart NGINX and check service status and then try to login again to Prometheus.

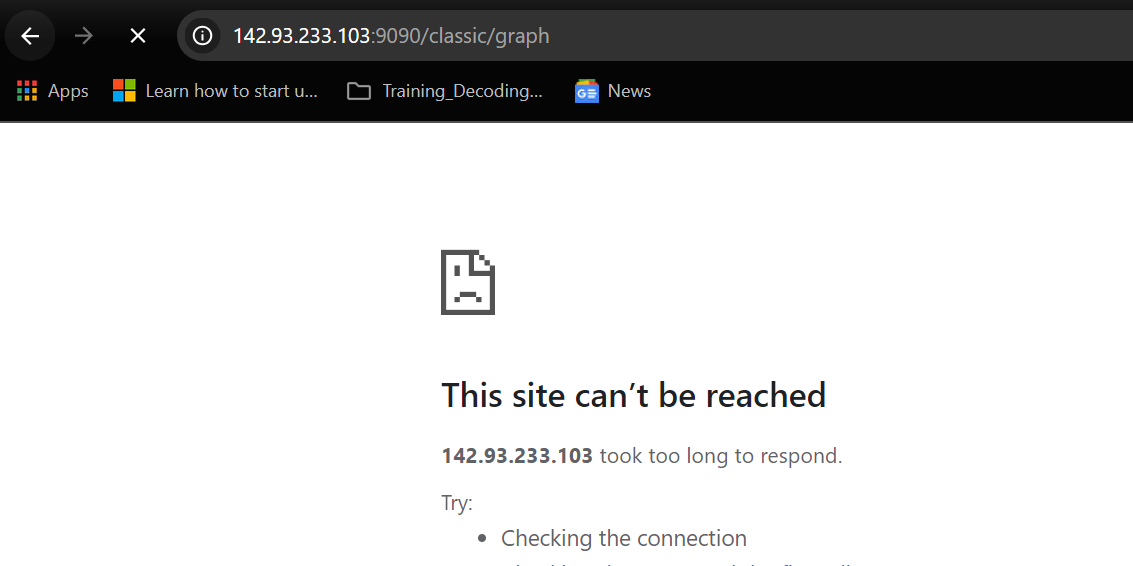


At this point anyone can still access prometheus without a password using directly the IP and port: <http://142.93.233.103:9090/classic/graph>

but we can block external connections using following commands:

| iptables -A INPUT -p tcp -s localhost --dport 9090 -j ACCEPT  iptables -A INPUT -p tcp --dport 9090 -j DROP  iptables -A INPUT -p tcp -s localhost --dport 9100 -j ACCEPT  iptables -A INPUT -p tcp --dport 9100 -j DROP  iptables -L |
| --- |

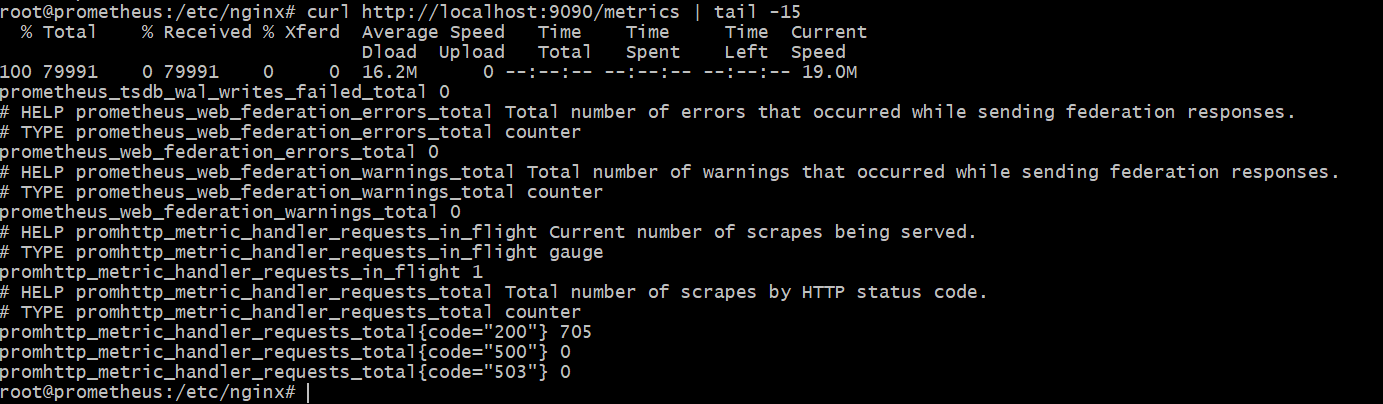




1. **Scrape Target Basics:**

We can see the metrics prometheus exposes by doing a curl to:

curl [http://localhost:9090/metrics](http://prometheus:9090/metrics)



We can also inspect on the server using:

| **config.file** | /etc/prometheus/prometheus.yml |
| --- | --- |

or review it from <https://prometheus1.testing-devops.info/classic/config>

1. **Prometheus Rules**

**Recording Rules:**

CD into prometheus folder and create a rules file:

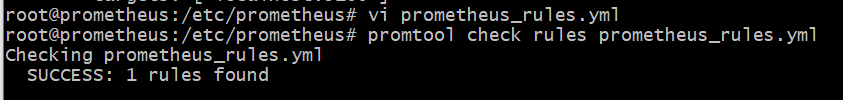
| cd /etc/prometheus  sudo vi prometheus\_rules.yml |
| --- |

Add a test expression as recording rule:

| groups:  - name: custom\_rules  rules:  - record: node\_memory\_MemFree\_percent  expr: 100 - (100 \* node\_memory\_MemFree\_bytes / node\_memory\_MemTotal\_bytes) |
| --- |

And also check if the rule file has good syntax using a tool called promtool:

| promtool check rules prometheus\_rules.yml |
| --- |



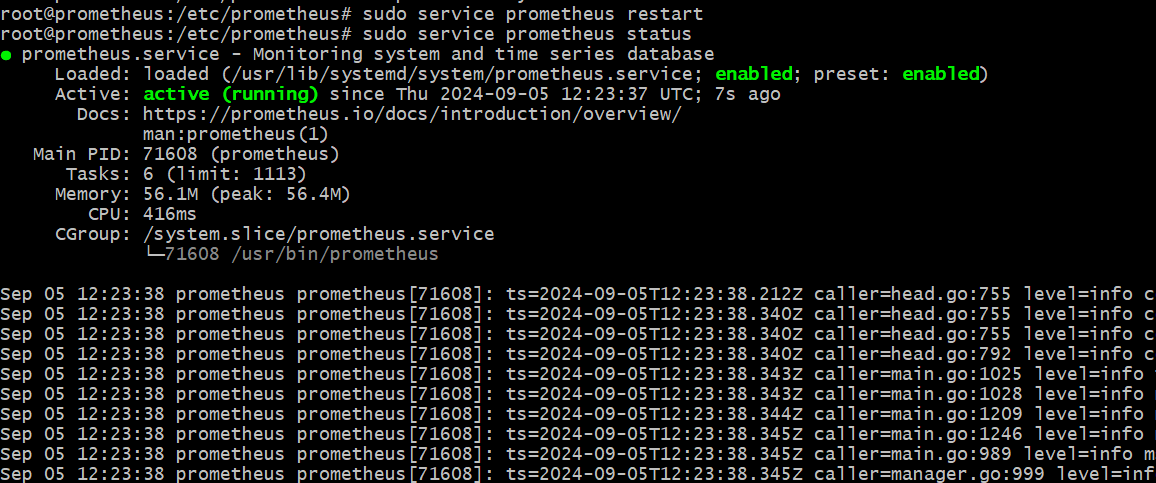
If we get a success result on the check, we can add the *prometheus\_rules.yml* reference to the *prometheus.yml* rule\_files section.

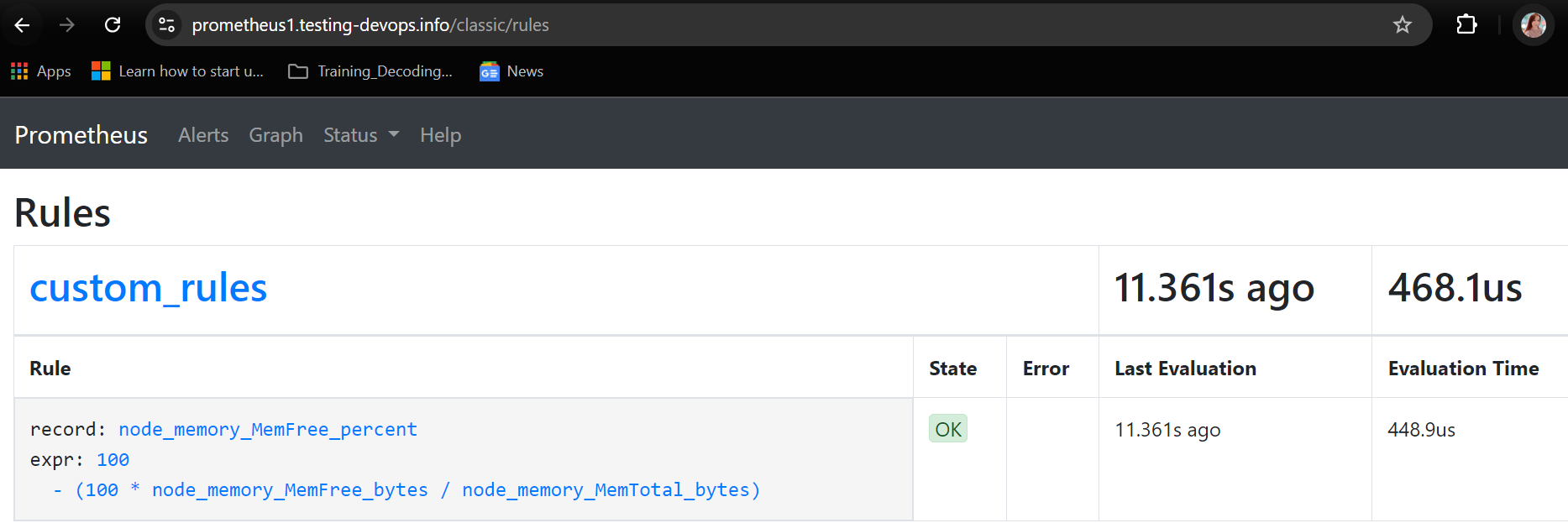
rule\_files:

- "prometheus\_rules.yml"

And then we can restart Prometheus service:

| sudo service prometheus restart  sudo service prometheus status |
| --- |





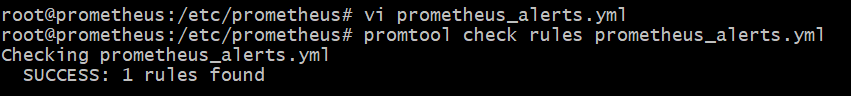
**Alerting rules:**

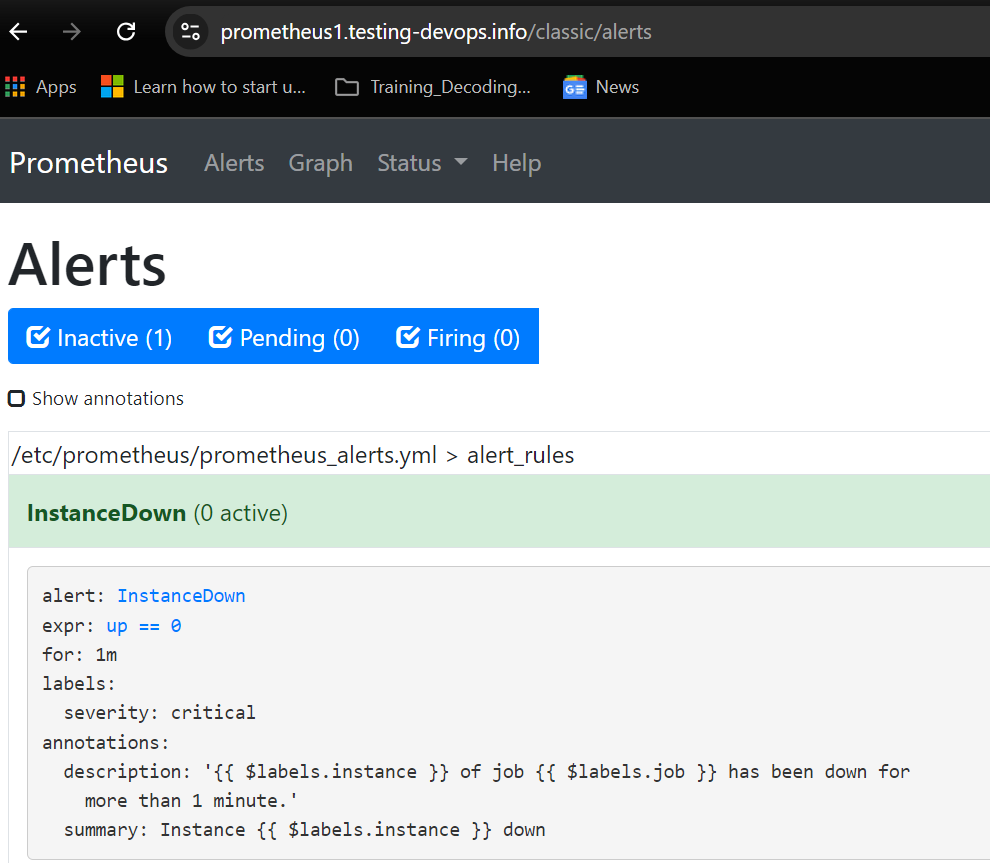
We create a different file and we need to add the reference to it in the *rule\_files* section in *prometheus.yml*

**root@prometheus:/etc/prometheus# cat prometheus\_alerts.yml**

| **groups:**  **- name: alert\_rules**  **rules:**  **- alert: InstanceDown**  **expr: up == 0**  **for: 1m**  **labels:**  **severity: critical**  **annotations:**  **summary: 'Instance {{ $labels.instance }} down'**  **description: '{{ $labels.instance }} of job {{ $labels.job }} has been down for more than 1 minute.'** |
| --- |

Similar as above section, review it using promtool and restart Prometheus service.



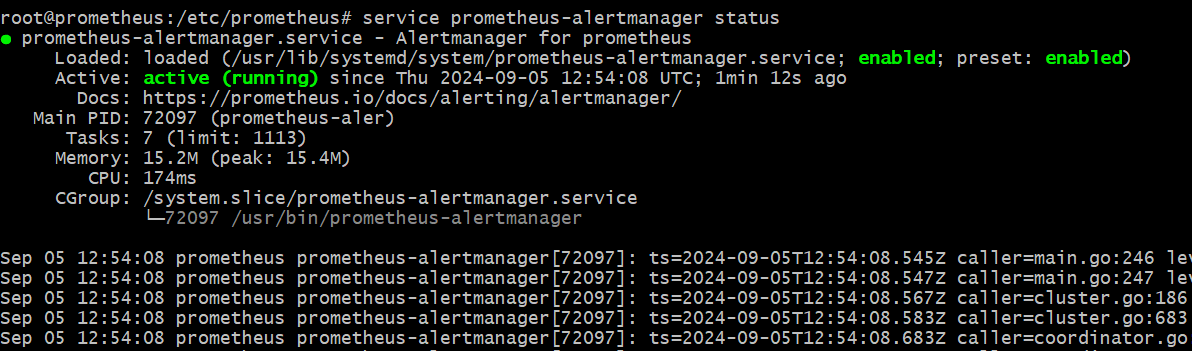


1. **Install Prometheus AlertManager:**

| sudo apt install prometheus-alertmanager |
| --- |

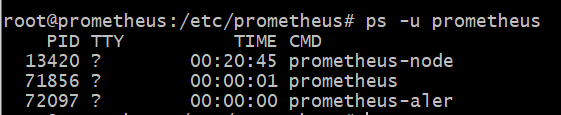
It has started a new service called prometheus-alertmanager

| sudo service prometheus-alertmanager status |
| --- |

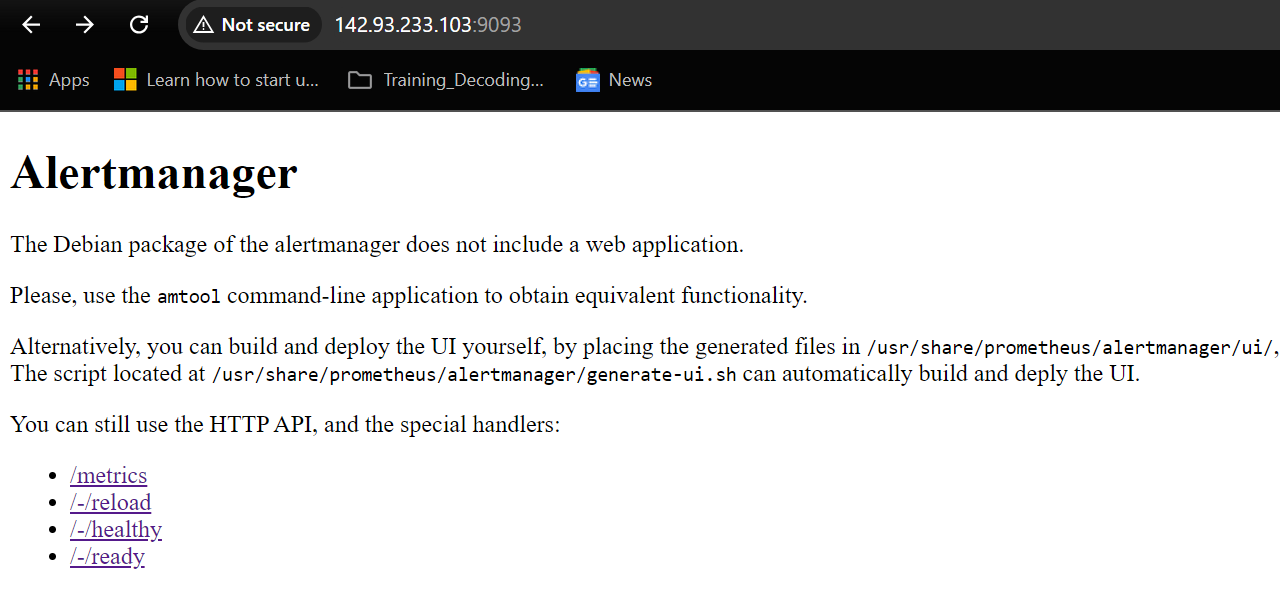


It is also managed by the user prometheus.

| ps -u prometheus |
| --- |

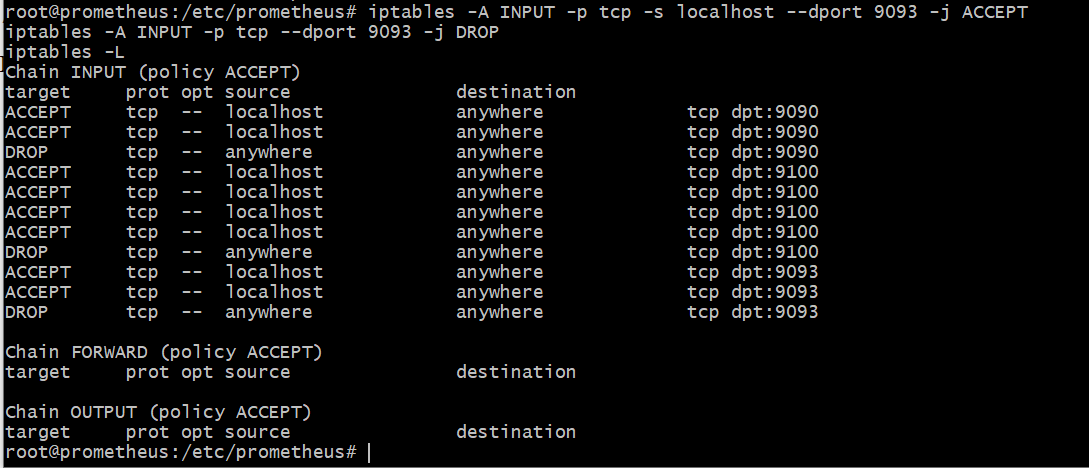


Visit http://[your domain name or IP]:9093/



Block port 9093 for external requests:

| iptables -A INPUT -p tcp -s localhost --dport 9093 -j ACCEPT  iptables -A INPUT -p tcp --dport 9093 -j DROP  iptables -L |
| --- |



**NOTE:**

| **iptables** settings will be lost in case of system reboot. You will need to reapply them manually, or install **iptables-persistent:**   | sudo apt install iptables-persistent | | --- |   This will save your settings into two files called,  /etc/iptables/rules.v4  /etc/iptables/rules.v6  Any changes you make to the **iptables** configuration won't be auto saved to these persistent files, so if you want to update these files with any changes, then use the commands,  iptables-save > /etc/iptables/rules.v4  iptables-save > /etc/iptables/rules.v6 |
| --- | --- |

Check the endpoint in the prometheus.yml is correctly set for the location of your alert manager.

| sudo vi /etc/prometheus/prometheus.yml |
| --- |

Mine is set to the alert manager running locally on localhost:9093

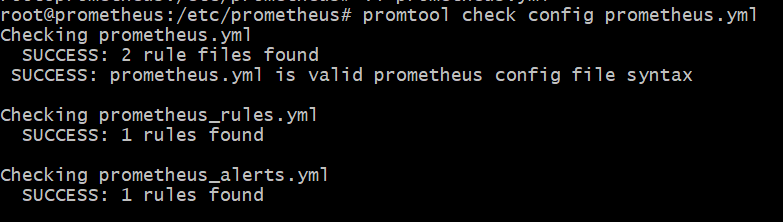
| ...  # Alertmanager configuration  alerting:  alertmanagers:  - static\_configs:  - targets: ['localhost:9093'] |
| --- |

We can optionally also add the alert manager metrics endpoint to be scraped by Prometheus as well so that we can monitor its performance.

| scrape\_configs:  ...  - job\_name: alertmanager  static\_configs:  - targets: ['localhost:9093'] |
| --- |

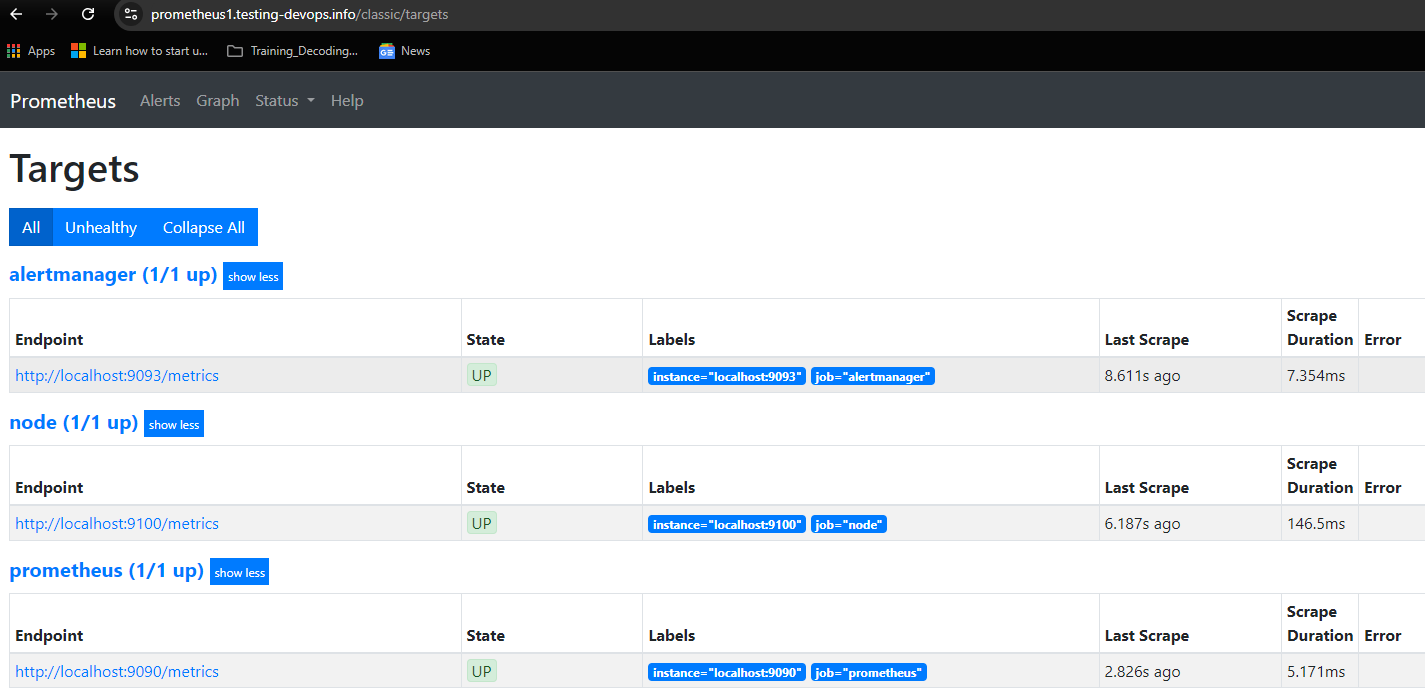
If you edit the prometheus.yml, remember to check it using promtool

| promtool check config /etc/prometheus/prometheus.yml |
| --- |



And then restart Prometheus:

| sudo service prometheus restart  sudo service prometheus status |
| --- |



1. **Install Grafana:**

Update package lists:

| sudo apt update |
| --- |

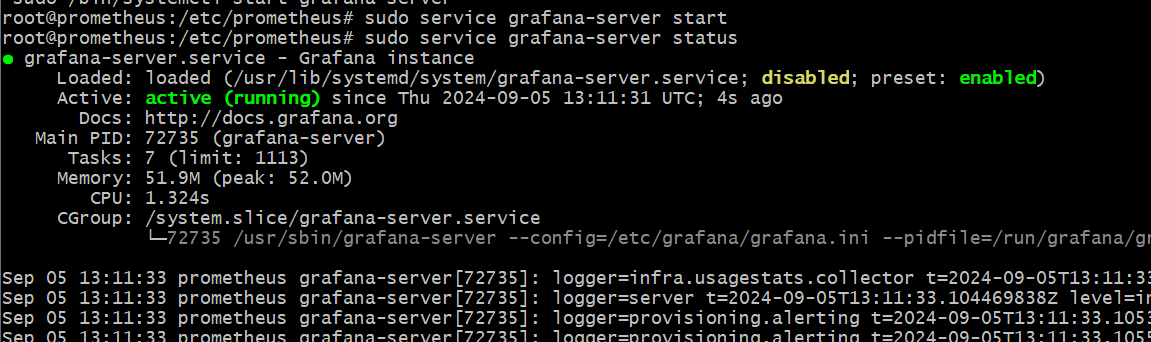
Install Grafana dependencies:

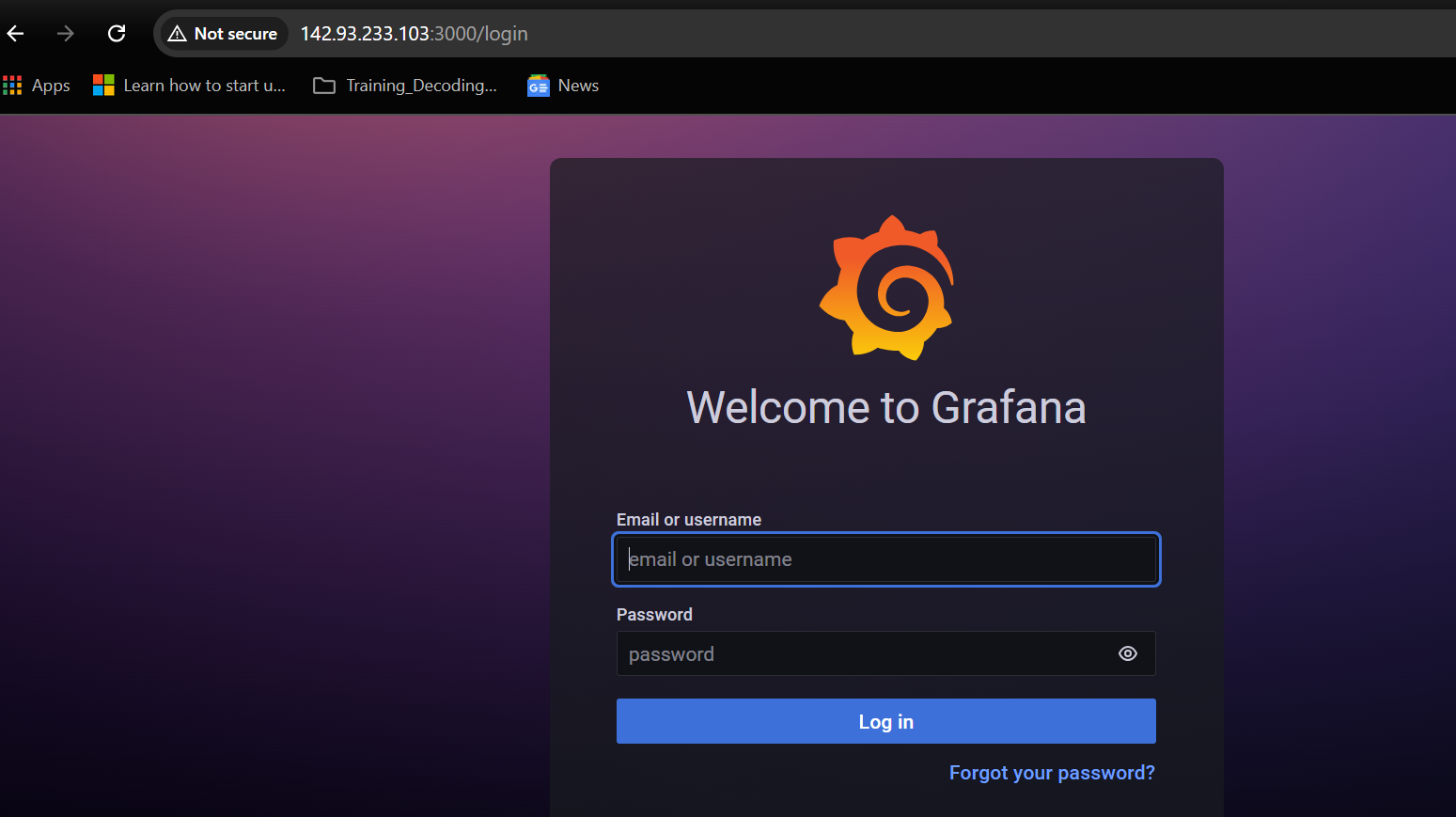
| sudo apt-get install -y adduser libfontconfig1 |
| --- |

Download the binary & run a Debian package manager:

| wget https://dl.grafana.com/oss/release/grafana\_9.3.2\_amd64.deb  sudo dpkg -i grafana\_9.3.2\_amd64.deb |
| --- |

Start & Check status of Grafana:





Your Grafana server will be hosted at: http://[your Grafana server IP]:3000

The default Grafana login is: Username : **admin** Password : **admin**