Grafana/Prometheus/Nginx

- Installation de Grafana
- Monitoring de Nginx sur Grafana

Installation de Grafana

Sources : https://www.d4d.lt/how-to-install-and-configure-grafana-and-prometheus-on-debian-12

1. Install the prerequisite packages:

```
apt-get install -y apt-transport-https software-properties-common wget
```

Monitoring de Nginx sur Grafana

https://blog.netmanageit.com/nginx-monitoring-with-grafana-prometheus-and-telegraf-howto/

Setup and Configure NGINX Stubs module and Telegraf on the endpoints to be monitored

Once you have completed the core install above, I will show you my custom setup method for installing and configuring telegraf on your NGINX servers / endpoints and enabling and configuring the NGINX Stubs module on a seperate site/port and limiting access to just the Monitoring Server we setup. Once we do this on one or more endpoints, we will move on to the prometheus.yml configuration and the Grafana prebuilt dashboard setup for NGINX monitoring.

Enable NGINX Stubs module

Go to /etc/nginx/sites-available and create a new site, lets call it "nginxstubs.conf"

□□
nano /etc/nginx/sites-available/nginxstubs.conf

Now paste this text into the file editor and save it. Note : change the allow directive to whitelist IP/subnet of your choice and deny the rest. This will protect the /nginx_status site on port 81 from being accessed by the public.

```
server {
        listen 81 default_server;
        listen [::]:81 default_server;

        root /var/www/html;
        index index.html index.htm index.nginx-debian.html;

        server_name _;

        location / {
                try_files $uri $uri/ =404;
        }

        location /nginx_status {
                stub_status;
                allow 127.0.0.1;
                allow 192.168.0.0/24;
                deny all;
                }
}
```

NGINX Stubs Sites-available Configuration

Next, lets enable the site

```
sudo ln -s /etc/nginx/sites-available/nginxstubs.conf /etc/nginx/sites-enabled/nginxstubs.conf
```

Now restart NGINX

```
sudo systemctl restart nginx
```

Install and Configure Telegraf

If your using most flavors of Debian/Ubuntu you can simple use the apt-get install command to install Telegraf.

□□
sudo apt-get install telegraf

If you get the message it cannot find the package for that particular distro or version use the script/command below to completely get and install it from official repo.

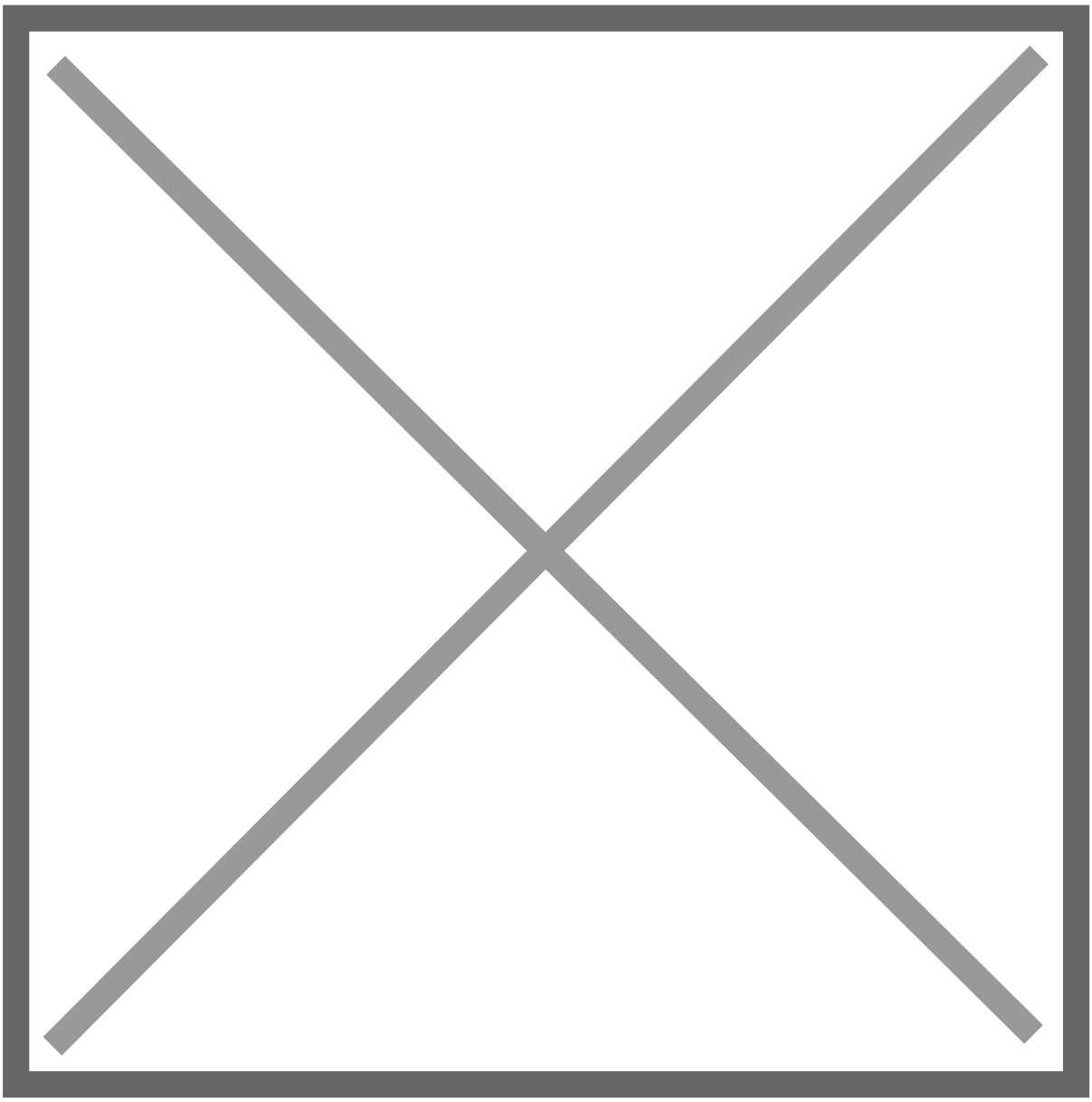
□□
curl --silent --location -O https://repos.influxdata.com/influxdata-archive.key && echo "943666881a1b8d9b849b74caebf02d3465d6beb716510d86a39f6c8e8dac7515 influxdata-archive.key" | sha256sum -c - && cat influxdata-archive.key | gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/influxdata-archive.gpg > /dev/null && echo 'deb [signed-by=/etc/apt/trusted.gpg.d/influxdata-archive.gpg] https://repos.influxdata.com/debian stable main' | sudo tee /etc/apt/sources.list.d/influxdata.list sudo apt-get update && sudo apt-get install telegraf

Once installed nagivate to the /etc/telegraf directory and edit the telegraf.conf file to add config code and enable prometheus monitoring.

□□
sudo nano /etc/telegraf/telegraf.conf

Past the following Code block at the top of the file

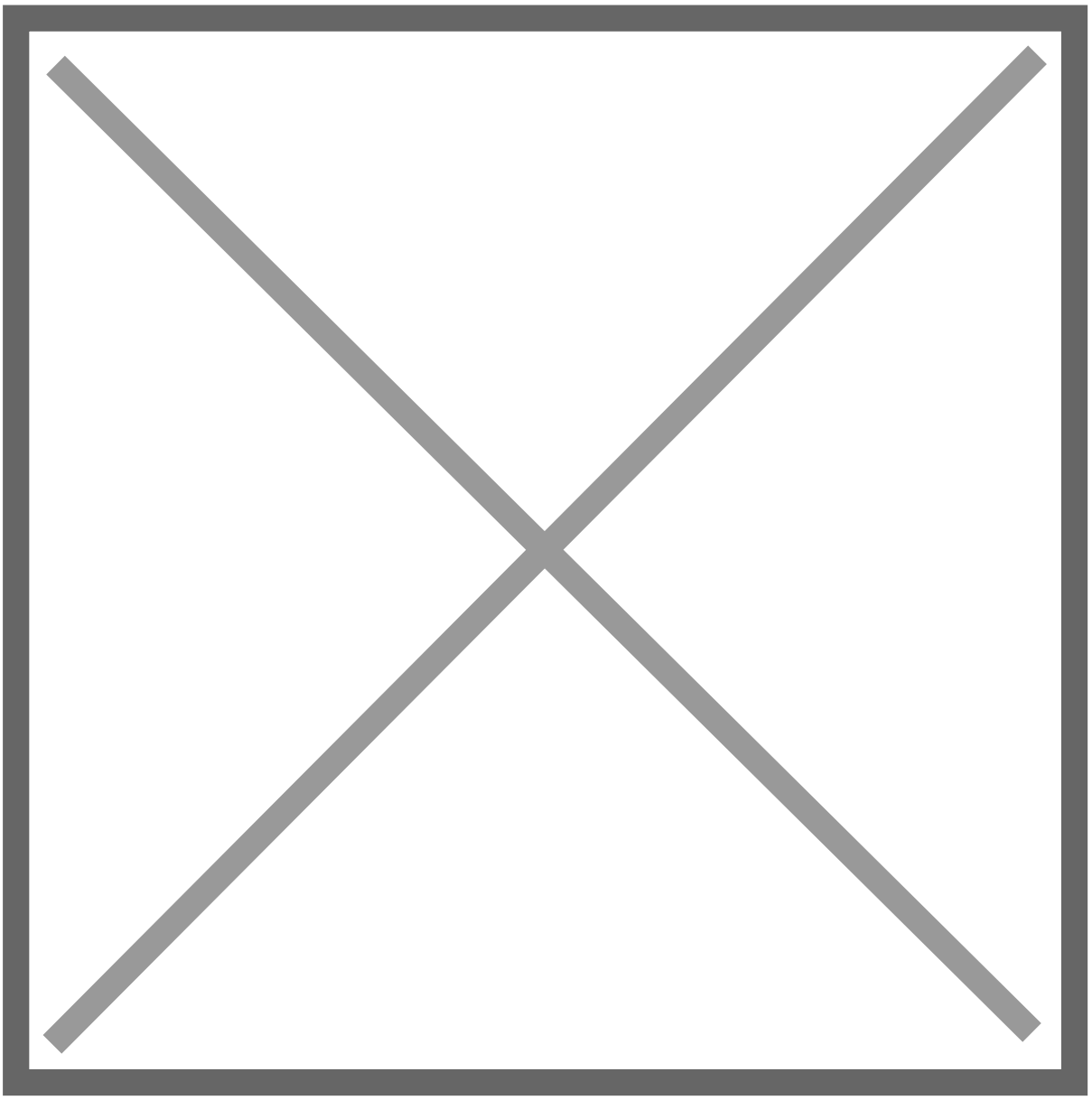
```
#nginx-metrics and logs
[[inputs.nginx]]
   □ urls = ["http://localhost/nginx_status"]
   □ response_timeout = "5s"
[[inputs.tail]]
  □ name_override = "nginxlog"
  □ files = ["/var/log/nginx/access.log"]
   □ from_beginning = true
  □ pipe = false
  □ data_format = "grok"
  □ grok_patterns = ["%{COMBINED_LOG_FORMAT}"]
```



Code Block pasted to configure proper parsing of NGINX logs

Now search for "prometheus" to get to the code section you need to modify last. This will show [[outputs.prometheus_client]] . Make sure this line is uncommented, along with listen directive.

Then change listen = "0.0.0.0:9125" Save file and exit



Telegraf Prometheus client configuration

Lastly, use one or more of the following commands to grant "_telegraf" service full permissions for the /var/log/nginx folder. This is because it takes additional metrics from the access.log and error.log file and combines it with the Stubs metrics for pretty good stats!

Add telegraf user to www-data

□□

sudo usermod –aG www-data _telegraf

NOTE: Certain Telegraf installs, the user is "telegraf" not "_telegraf". If the command above doesn't work substitute "telegraf" at the end of command.

Change Nginx Log directory owner to www-data recursively to allow telegraf to access to Nginx log files

```
sudo chown -R www-data:www-data /var/log/nginx
```

Restart Telegraf

```
sudo systemctl restart telegraf
```

At this point your endpoint is now advertising metrics through telegraf on your endpoint at IP:9125 . Telegraf reads the stubs metrics from IP:81/nginx_status.

Configure Server Prometheus Scrape Jobs and Import Prebuild Grafana Dashboard

Next step is to configure the prometheus scrape jobs, I have included a template for you to fill in for each of the sites you configured above.

Below you see Node_Exporter section, target field enter the IP of your grafana server. Then the format for the job names, each job name is for each additional endpoint you configured to monitor NGINX. The job name can be whatever and is not referenced in the dashboard.

What IS important is the targets: section, this is the endpoint IP of each server you congigured. Next important note is that to get the correct label in the Dashboard, i used a relabel function. Simply change the name of 'ENDPOINT1NAME' and replacement: ENDPOINT1NAME with a user friendly name of your choice.

This is a good additional configuration, as without it, prometheus will default to using the IP:port for the instance ID. The more instances you have, the harder it is to identify. You will see what I mean further on down below the article.

NOTE: If you want additional configurations, simple copy the last job code block starting with "- job name, and ending with replacement: "

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

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

  - job_name: nginxendpoint1
    # If telegraf is installed, grab stats about the local
    # machine by default.
    static_configs:
      - targets: ['IPOFENDPOINT1:9125']
        labels:
          instance: 'ENDPOINT1NAME'
    relabel_configs:
      - source_labels: [__address__]
        target_label: instance
        replacement: ENDPOINT1NAME
  - job_name: nginxendpoint2
    # If telegraf is installed, grab stats about the local
    # machine by default.
    static_configs:
      - targets: ['IPOFENDPOINT2:9125']
        labels:
          instance: 'ENDPOINT2NAME'
    relabel_configs:
      - source_labels: [__address__]
        target_label: instance
        replacement: ENDPOINT2NAME
```

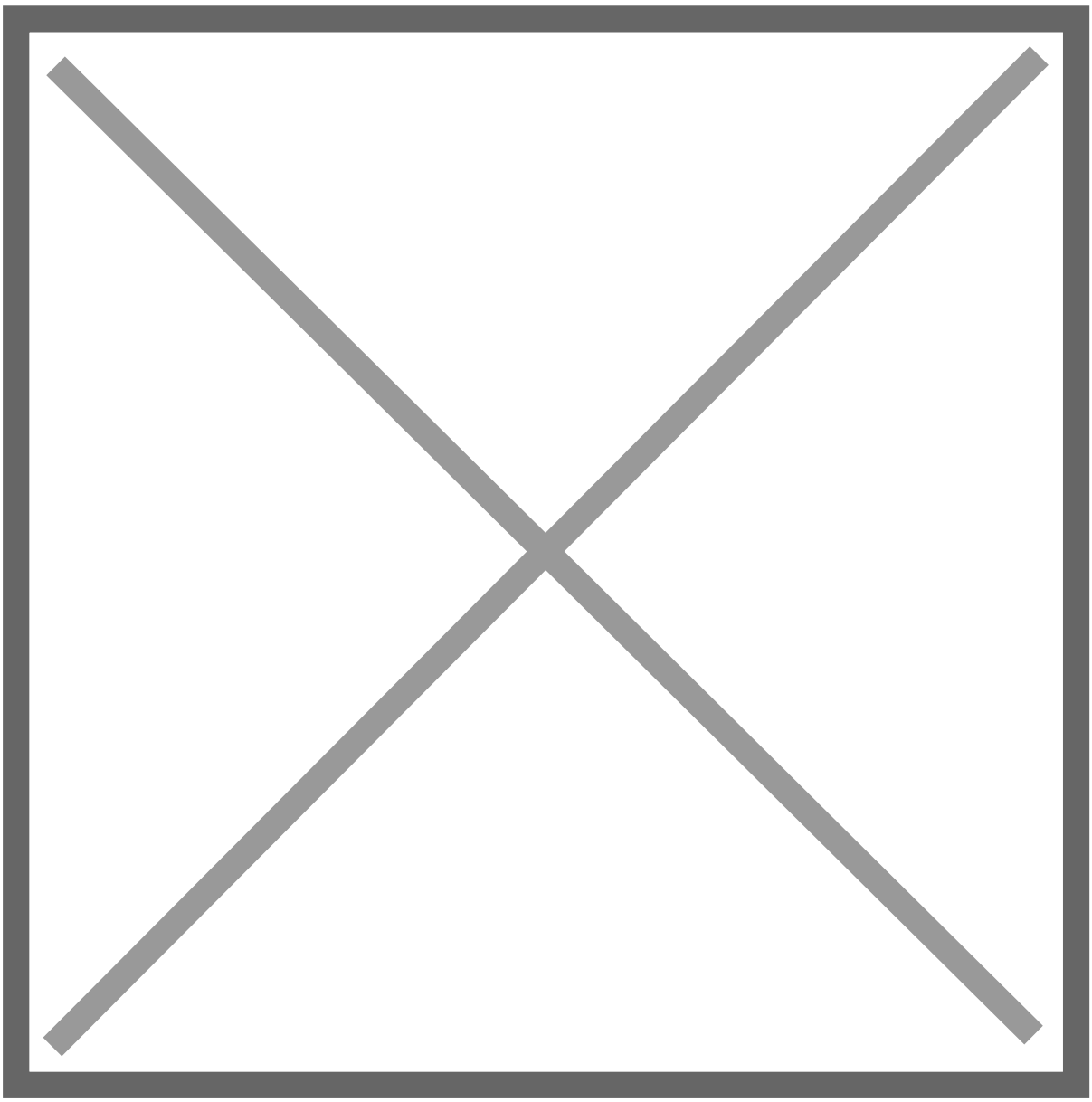
Prometheus.yml Scrape Job Configuration template

Save the file and restart prometheus on the server

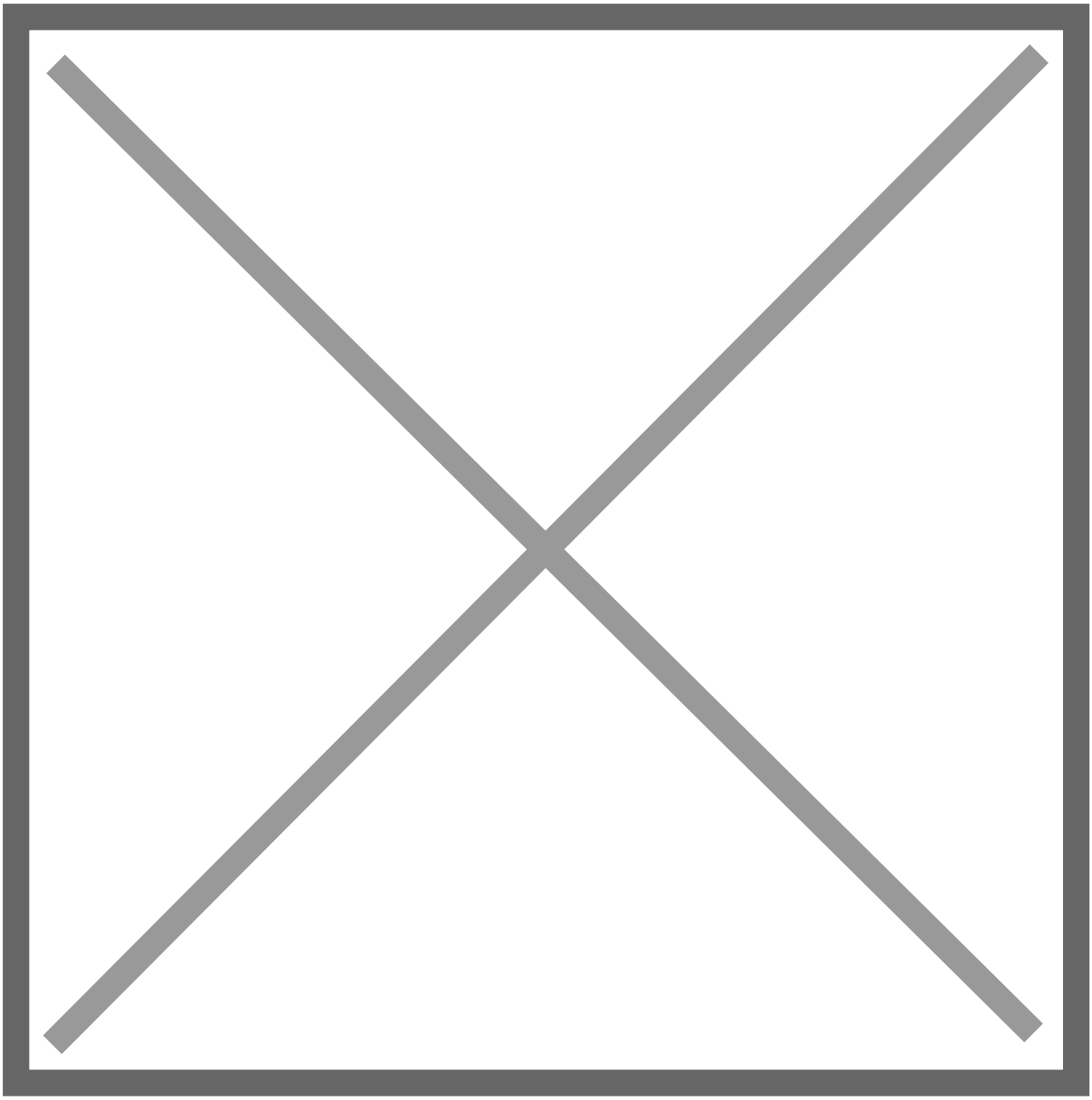
□□
sudo systemctl restart prometheus

Install / Import the Prebuilt Grafana dashboard for the configuration above

All we need to do to view the NGINX metrics data, and switch between instances we configured above. Go to Dashboards, new and import.

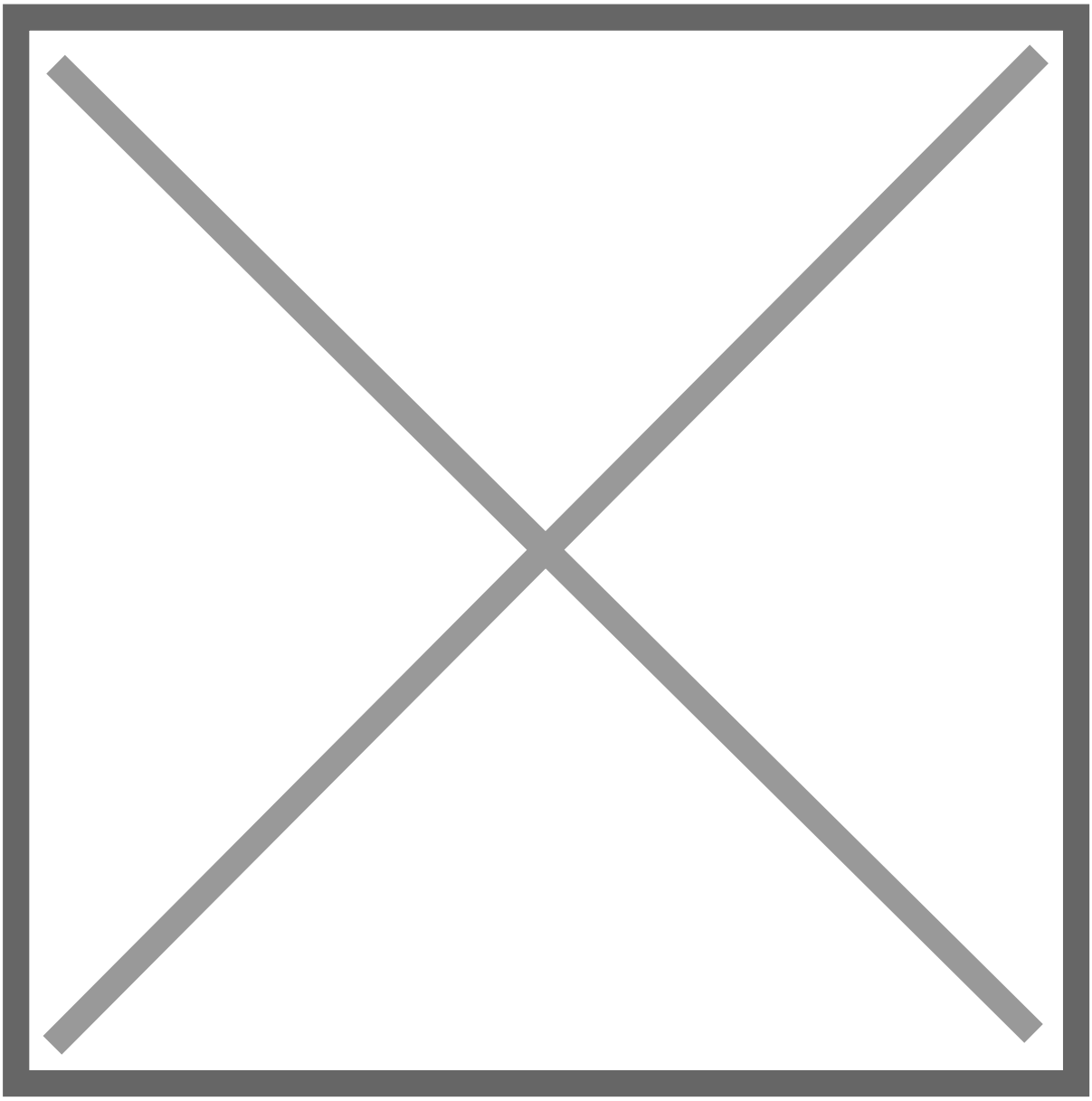


Enter Dashboard ID - "14900" and then load and finish up. You should now see the NGINX monitoring dashboard in your dashboards list.



ID 14900 automatically pulls the dashboard JSON from the Grafana library online

Click on it, and you can then mofidy the layout as you wish. Notice the following important top selection boxes. The only one you need to change is the "Instance" selector dropdown box. Notice all the friendly names of your configured instances we made prior in the prometheus.yml scrape configuration for each endpoint.



You can see the red arrow pointing to the instance selector dropdown for different NGINX Server Names