

## BAB V

### IMPLEMENTASI DAN PENGUJIAN

pada bab ini akan dilakukan implementasi serta melakukan pengujian terhadap penerapan sistem *dashboard* untuk *monitoring* sistem dan jaringan berbasis *Cacti* yang menggunakan perangkat lunak *Grafana*.

#### 5.1 Instalasi dan Konfigurasi Software

Pada instalasi dan konfigurasi *software* ini akan membahas langkah-langkah instalasi dan konfigurasi *Cacti*, *Grafana*, *Pushgateway*, *Prometheus*.

##### 5.1.1 Instalasi dan Konfigurasi Cacti

- Sebelum melakukan instalasi terlebih dahulu lakukan update dan upgrade sistem.

```
$ sudo apt update && sudo apt upgrade
```

- Karena *Cacti* memerlukan *mysql database* maka kita harus menginstalnya, disini kita menggunakan *mariadb database*.

```
$ sudo apt install -y mariadb-server
```

- Lakukan instalasi *mariadb* menggunakan perintah berikut. Jangan lupa untuk melakukan set *root password*.

```
$ sudo mysql_secure_installation
```

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current password for the root user. If you've just installed MariaDB, and you haven't set the root password yet, the password will be blank, so you should just press enter here.

```
Enter current password for root (enter for none): [enter]  
OK, successfully used password, moving on...
```

Setting the root password ensures that nobody can log into the MariaDB root user without the proper authorisation.

Set root password? [Y/n] **Y**

New password:

Re-enter new password:

Password updated successfully!

Reloading privilege tables..

... Success!

By default, a MariaDB installation has an anonymous user, allowing anyone to log into MariaDB without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

Remove anonymous users? [Y/n] **Y**

... Success!

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] **Y**

... Success!

By default, MariaDB comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] **Y**

- Dropping test database...

... Success!

- Removing privileges on test database...

... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] **Y**

... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB installation should now be secure.

Thanks for using MariaDB!

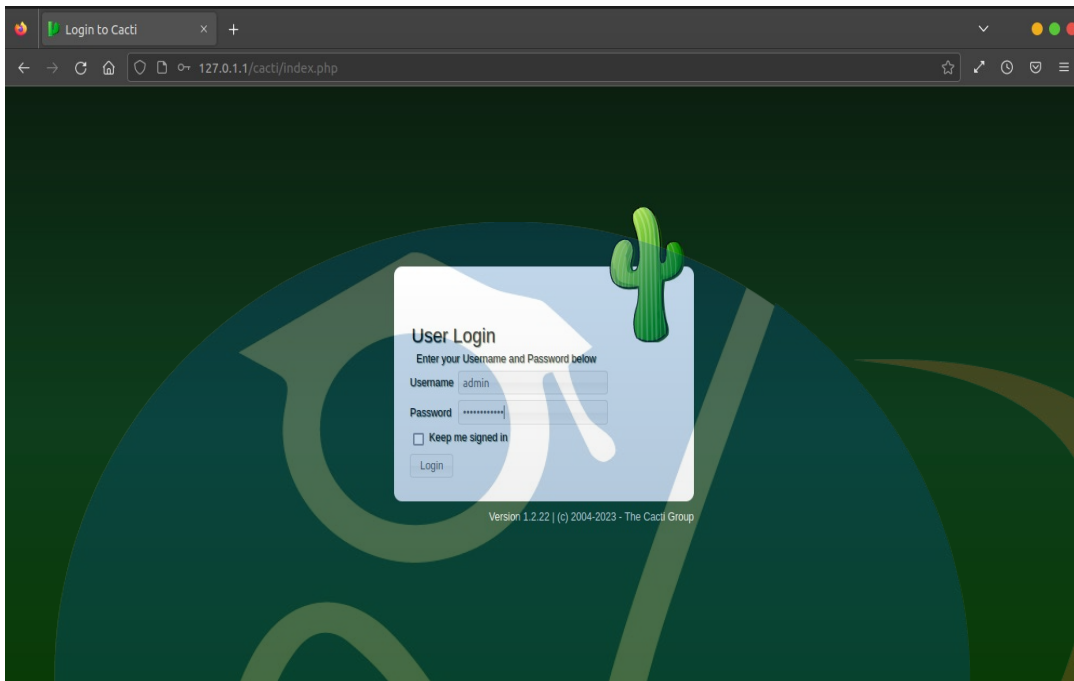


Otherwise, you should probably choose this option.	
Configure database for cacti with dbconfig-common?	
Yes	No

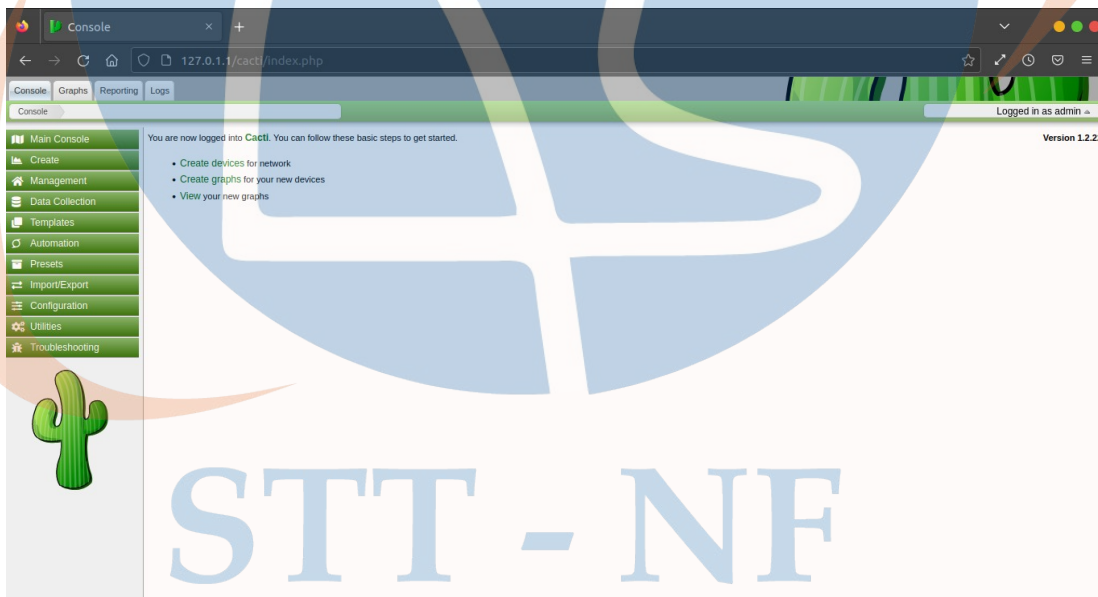
- Buat *password database* untuk *Cacti*. *Password* yang kita buat dan masukkan menjadi *password akun database*, ini juga akan menjadi *password login* menuju *Cacti dashboard* nantinya.

Package configuration	
Configuring cacti	
Please provide a password for cacti to register with the database server. If left blank, a random password will be generated.	
MySQL application password for cacti:	
*****	
Ok	Cancel

- Konfirmasi *password*.
- Tunggu proses instalasi selesai. Setelah proses instalasi selesai kita dapat melakukan akses menuju *dashboard Cacti* dengan cara melakukan akses "*http://ip-server/cacti*". Untuk login *default* username adalah *admin* dan *password* adalah kita bisa buat saat melakukan instalasi *Cacti* sebelumnya.



Gambar 5.3 Tampilan Login Cacti



Gambar 5.4 Tampilan Cacti Dashboard

- Sampai disini kita sudah berhasil melakukan instalisasi *Cacti monitoring tools* pada Ubuntu server 20.04 LTS.

### 5.1.2 Instalasi dan Konfigurasi Grafana

- Instal *Grafana* bisa mengambil dari repositori resmi *Grafana*. Unduh *GPG key* dan tambahkan. Ketik dengan perintah di terminal ubuntu.

```
$ wget -q -O - https://packages.grafana.com/gpg.key | sudo apt-key add -
```

- Setelah itu tambahkan ke repositori *Grafana*.

```
$ sudo add-apt-repository "deb https://packages.grafana.com/oss/deb stable main"
```

- Setelah ditambahkan, kita perlu memperbarui *cache* ART dan melanjutkan instalasi *Grafana*.

```
$ sudo apt update
```

- Instal *Grafana* di terminal setelah sudah memperbarui atau mengupdate *cache* ART.
- Setelah instalasi, tambahkan *Grafana* ke autostart dan jalankan daemon *Grafana* itu sendiri.

```
$ sudo systemctl enable grafana-server
```

```
$ sudo systemctl start grafana-server
```

- Instalasi repositori *Grafana* sudah selesai dan siap digunakan. Terakhir, periksa status *Grafana*.

```
$ sudo systemctl status grafana-server
```

- Port *http default Grafana* adalah 3000, kita harus mengizinkan akses ke *port* ini di *firewall*. Ubuntu hadir dengan *firewall ufw*.

```
$ sudo apt -y install ufw
```

- Kemudian aktifkan layanan *firewall*.

```
$ sudo ufw enable
```

- Buka port di *firewall*.

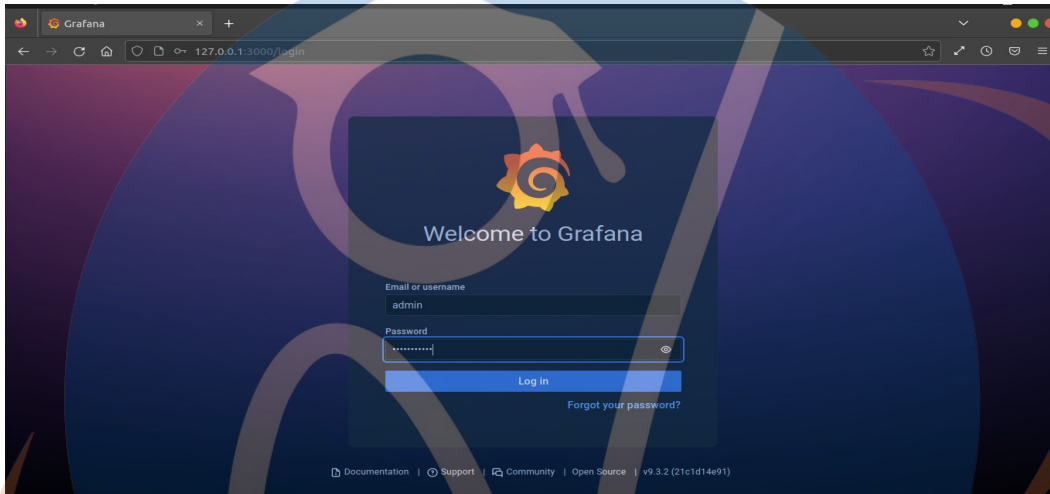
```
$ sudo ufw allow ssh
```

```
$ sudo ufw allow 3000/tcp
```

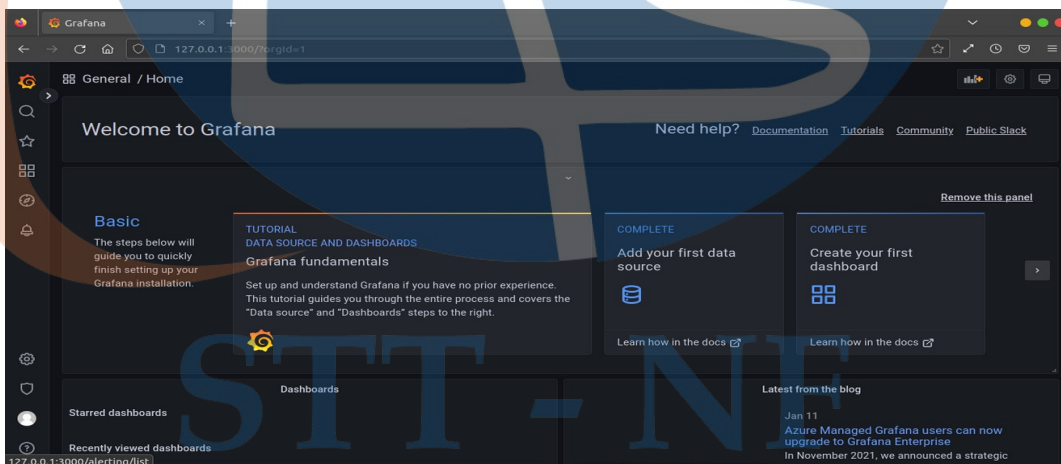
- Untuk mengizinkan akses hanya dari *subnet* tertentu, gunakan

```
$ sudo ufw allow from 127.0.0.1/1 to any port 3000
```

- Akses dasbor *Grafana* menggunakan alamat IP *server* atau nama *host* dan *port* 3000.



Gambar 5.5 Tampilan Login Grafana



Gambar 5.6 Tampilan Dashboard Grafana

### 5.1.3 Instalasi dan Konfigurasi Prometheus

*Prometheus* adalah *open source* sistem *monitoring* dan *alerting toolkit* yang awalnya dikembangkan oleh *SoundCloud*. *Prometheus* bergabung dengan *Cloud Native Computing Foundation* pada tahun 2016. *Prometheus* memiliki banyak *metric* yang dibutuhkan untuk memantau sistem dan dapat dikombinasikan dengan *Grafana* untuk visualisasi.

- Membuat *user* untuk *Prometheus*.

```
$ useradd --no-create-home --shell /bin/false prometheus
```

- Membuat folder *Prometheus*.

```
$ mkdir /etc/prometheus
```

```
$ mkdir /var/lib/prometheus
```

- *Setting user* dan *group ownership* folder.

```
$ chown prometheus:prometheus /etc/prometheus
```

```
$ chown prometheus:prometheus /var/lib/prometheus
```

- *Download* dan *extract Prometheus*.

```
$ curl -LO https://github.com/prometheus/prometheus/releases/download/v2.31.0/prometheus-2.31.0.linux-amd64.tar.gz
```

```
$ tar xzvf prometheus-2.31.0.linux-amd64.tar.gz
```

- *Copy folder Prometheus*.

```
$ cp prometheus-2.31.0.linux-amd64/prometheus /usr/local/bin
```

- *Copy folder consoles* dan *console\_libraries*.

```
$ cp -r prometheus-2.31.0.linux-amd64/consoles /etc/prometheus
```

```
$ cp -r prometheus-2.31.0.linux-amd64/console_libraries /etc/prometheus
```

- *Setting user* dan *group ownership* folder *consoles* dan *consoles\_libraries*.

```
$ chown -R prometheus:prometheus /etc/prometheus/consoles
```

```
$ chown -R prometheus:prometheus /etc/prometheus/console_libraries
```

- Membuat file konfigurasi untuk *Prometheus*.



```
$ nano /etc/prometheus/prometheus.yml
```

- Isi dari file *prometheus.yml*

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

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
--config.file /etc/prometheus/prometheus.yml \
--storage.tsdb.path /var/lib/prometheus/ \
--web.console.templates=/etc/prometheus/consoles \
--web.console.libraries=/etc/prometheus/console_libraries

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

- Terakhir, ubah pemilik direktori ini menjadi pengguna yang dibuat sebelumnya dan *group Prometheus*.

```
$ chown -R prometheus:prometheus /var/lib/prometheus/
```

- Lalu aktifkan *Prometheus service*.

```
$ systemctl daemon-reload
```

```
$ systemctl enable prometheus
```

```
$ systemctl start prometheus
```

```
$ systemctl status prometheus
```

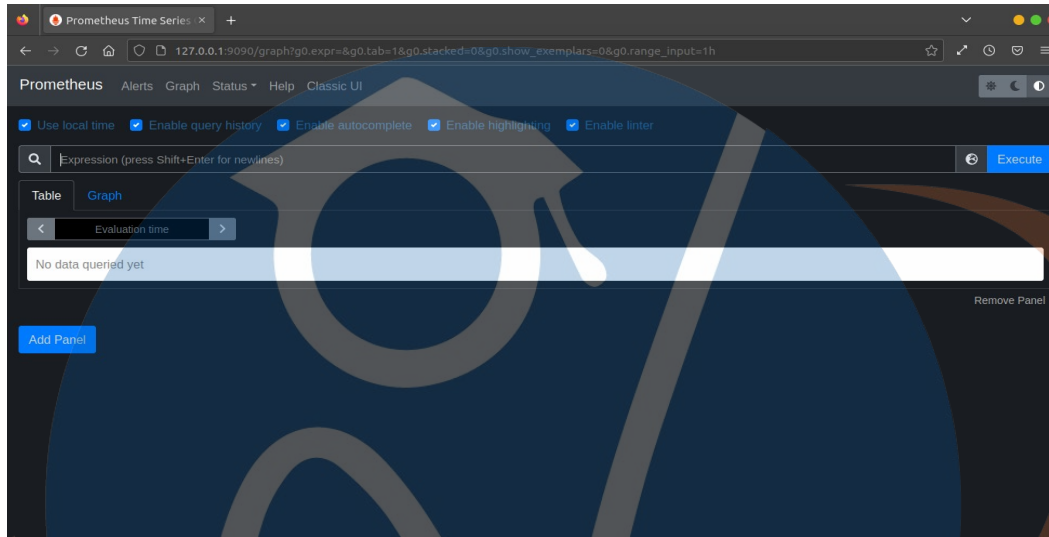
- Sekarang kita harus memastikan *firewall* dikonfigurasi dengan benar dan mengizinkan lalu lintas [ada port HTTPS(443), HTTP(80), dan 9090. *Server web nginx* mempresentasikan dirinya sebagai layanan *ufw*.

```
$ ufw allow in "Nginx Full"
```

```
$ ufw allow 9090/tcp
```

- Akses antarmuka web *Prometheus*. Secara standar, *Prometheus* tersedia di port HTTP 9090. Untuk memriksanya, membuka browser, dan tulis alamat IP *host*

tempat menginstal *Prometheus* sebelumnya yang telah dibuat. Jika semuanya sudah benar, maka akan membuka tab *dashboard Prometheus*. 'http://127.0.0.1:9090'.



Gambar 5.7 Tampilan Prometheus

#### 5.1.4 Instalasi dan Konfigurasi Pushgateway

Model tarikan *Prometheus* menyediakan metode yang sangat tangguh untuk mengumpulkan *metric*. Namun, ada beberapa kasus di mana model ini tidak cocok. *Prometheus Pushgateway* berfungsi sebagai perantara, menyediakan metode berbasis *push* untuk mengumpulkan metrik tanpa mengorbankan kesederhanaan *server Prometheus*.

- Membuat *user* dan *group* untuk *Pushgateway*.

```
$ sudo useradd -M -r -s /bin/false pushgateway
```

- Download dan instal *binary Pushgateway*.

```
$ wget  
https://github.com/prometheus/pushgateway/releases/download/v1.2.0/pushgateway-1.2.0.linux-  
amd64.tar.gz
```

```
$ tar xvfz pushgateway-1.2.0.linux-amd64.tar.gz
```

```
$ sudo cp pushgateway-1.2.0.linux-amd64/pushgateway /usr/local/bin/
```

```
$ sudo chown pushgateway:pushgateway /usr/local/bin/pushgateway
```

- Membuat file unit sistem untuk *Pushgateway*.

```
$ sudo vi /etc/systemd/system/pushgateway.service
```

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

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

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

- Mulai dan aktifkan layanan *Pushgateway*.

```
$ sudo systemctl enable pushgateway
```

```
$ sudo systemctl start pushgateway
```

- Verifikasi layanan berjalan dan menyajikan *metric*.

```
$ sudo systemctl status pushgateway
```

```
$ curl localhost:9091/metrics
```

- Konfigurasi *Pushgateway* sebagai target *scrape* untuk server *Prometheus*.  
*edit konfigurasi Prometheus.*

```
$ sudo vi /etc/prometheus/prometheus.yml
```

- Dibawah bagian *scrape\_configs*, tambahkan konfigurasi *scrape* untuk *Pushgateway*. Pastikan untuk menetapkan *honor\_labels:true*.

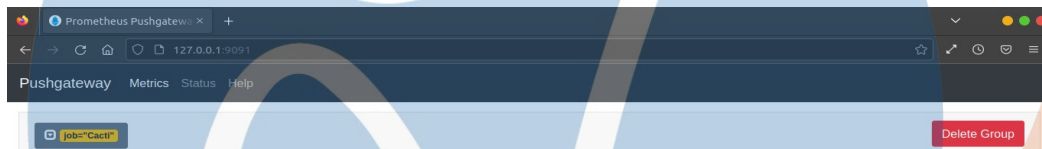
```
- job_name: 'Pushgateway'
```

```
honor_labels: true
static_configs:
- targets: ['localhost:9091']
```

- *Restart Prometheus* untuk memuat konfigurasi baru.

```
$ sudo systemctl restart prometheus
```

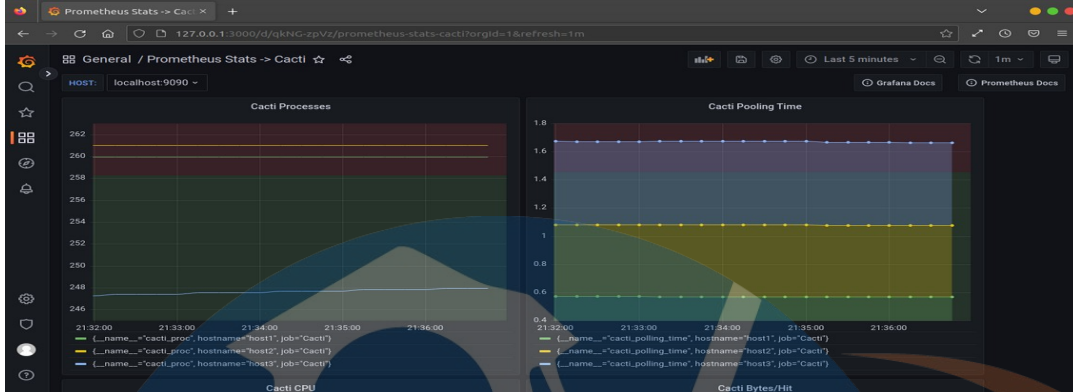
Akses antarmuka web *Prometheus*. Secara standar, *Prometheus* tersedia di port HTTP 9091. Untuk memriksanya, membuka browser, dan tulis alamat IP *host* tempat menginstal *Prometheus* sebelumnya yang telah dibuat. Jika semuanya sudah benar, maka akan membuka tab *dashboard Pushgateway*. '*http://127.0.0.1:9091*'.



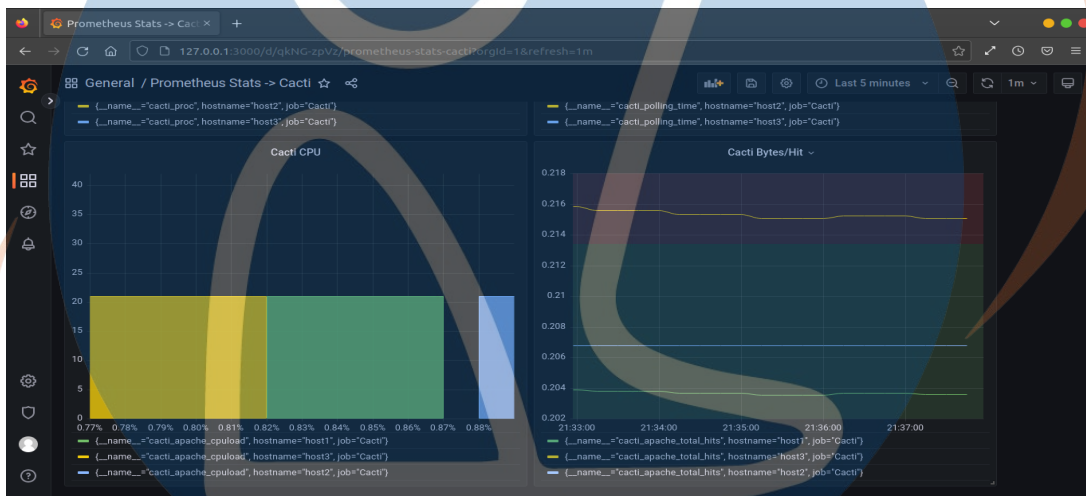
Gambar 5.8 Tampilan Pushgateway

## 5.2 Implementasi

Implementasi dilakukan berdasarkan analisis dan perancangan yang telah penulis buat. Selanjutnya akan dilakukan implementasi untuk mengetahui rancangan dan efektifitas sistem yang telah dibangun. Gambar berikut adalah tampilan dari *dashboard monitoring Grafana*. Sistem yang telah dibangun pada gambar berikut ini adalah tampilan dari sistem *dashboard*.



Gambar 5.1 Tampilan Sistem Dashboard



Gambar 5.2 Tampilan Sistem Dashboard

1. Panel Cacti Pooling Time (menggunakan visualisasi *Time Series*), pada *dashboard Cacti pooling time* merupakan tampilan transaksi data yang dilakukan dalam waktu yang sebenarnya (sekarang dan saat ini) merupakan dari sistem *pooling* yang terjadi. Data yang kita dapatkan mungkin sebenarnya sudah ada terlebih dahulu dan baru diproses saat data tersebut akan diminta.
2. Panel Cacti Processes (menggunakan visualisasi *Time Series*), *dashboard Cacti processes* merupakan *proses* kinerja yang sedang berjalan oleh sebuah *device* yang di *monitoring*.

3. Panel Cacti CPU (menggunakan visualisasi *histogram*), *dashboard Cacti cpu* merupakan proses yang sedang berjalannya *device CPU* yang sedang di *monitoring*.
4. Panel Cacti Bytes/Hit (menggunakan visualisasi *Time Series*), *dashboard Cacti bytes/hit* merupakan informasi digital komputasi dan telekomunikasi jumlah bit yang digunakan dalam proses *i/o*.

### 5.3 Pengujian

Pada tahap ini akan dilakukan pengujian sistem berdasarkan rancangan pengujian, pada tahap ini dilakukan implementasi *monitoring Cacti* menggunakan *Grafana* sebagai *dashboard* dan *Pushgateway prometheus* sebagai *transferrmasi* untuk mengakses *data* dari *Cacti* ke *Grafana*.

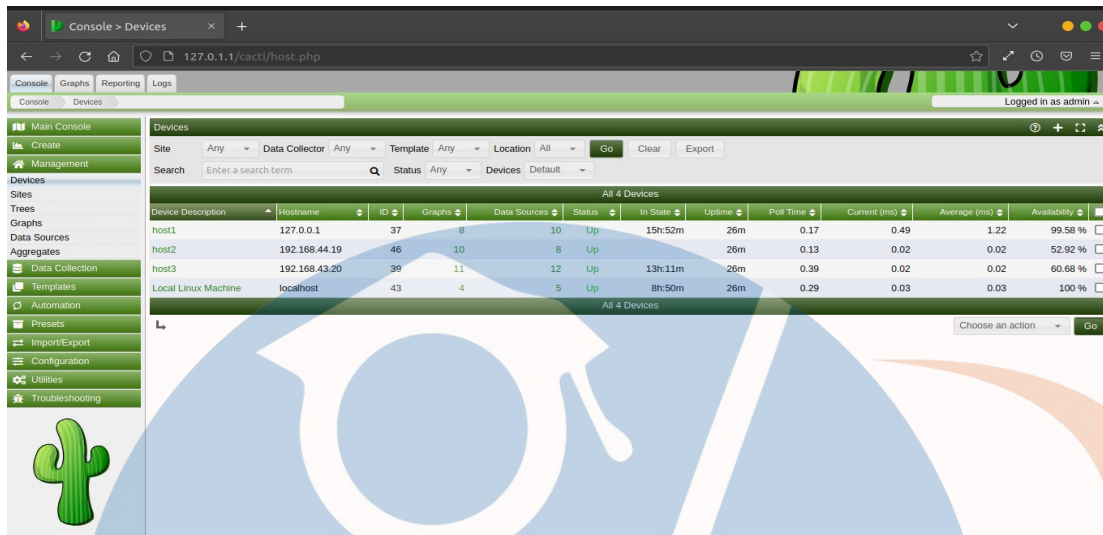
#### 5.3.1 Cacti

Menambahkan *device* di *Cacti* dan memonitoring *device tersebut*. Yang akan ditambahkan ada 3 IP Address yaitu '*Host (1) 127.0.0.1*', '*Host (2) 192.168.44.19*', '*Host (3) 192.168.43.20*'.



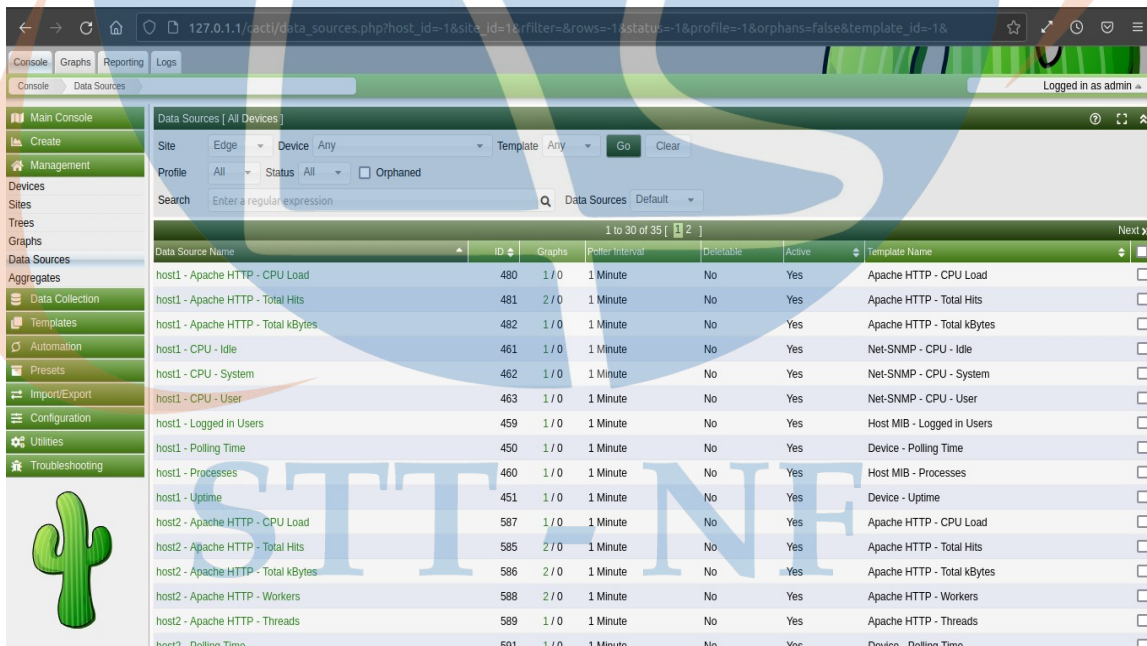
Gambar 5.9 Tampilan Home Cacti

Tampilan gambar diatas adalah tampilan *console web home Cacti*, yang dimana akan menambahkan *device* untuk di *monitoring*.



Gambar 5.10 Tampilan add Device Cacti

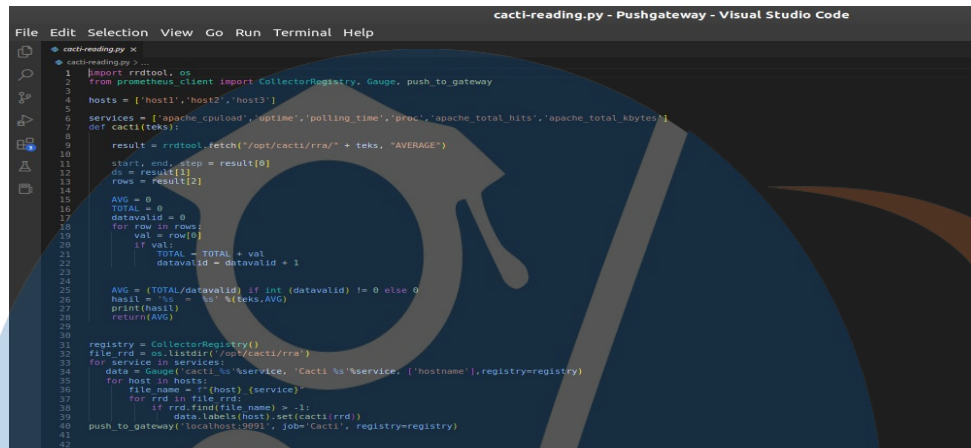
Gambar diatas adalah tampilan yang sudah ditambahkan *Host 1*, *Host 2*, dan *Host 3*. *Host* tersebut yang dimana nantinya akan di *monitoring* oleh *Cacti* dan juga implementasinya akan dibuatkan *dashboard* oleh *Grafana*.



Gambar 5.11 Tampilan Data Source Bertambah

### 5.3.2 Scripts Python

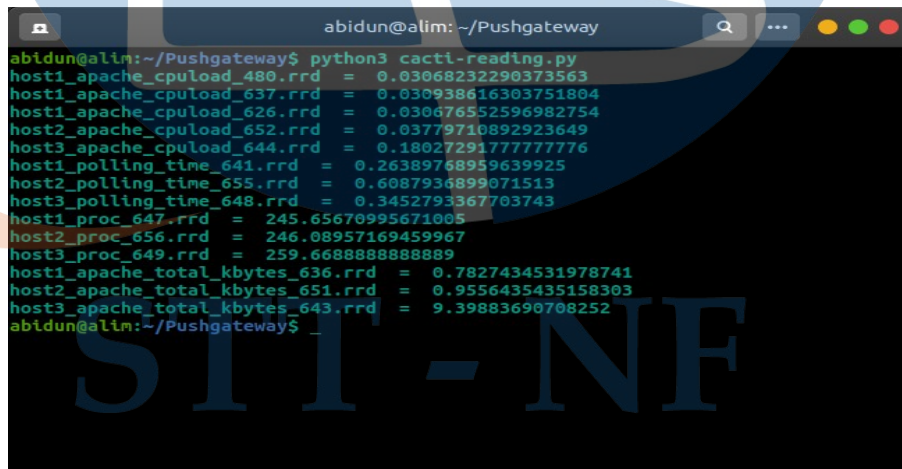
Scripts *python* ini dibuat untuk mengambil *data device Cacti* yang sudah dibuat di *Cacti* dan akan di *push* ke dalam *Pushgateway* dan juga *Prometheus*.



```
File Edit Selection View Go Run Terminal Help
cacti-reading.py x
cacti-reading.py 2...
1 import rrdtool, os
2 from prometheus_client import CollectorRegistry, Gauge, push_to_gateway
3
4 hosts = ['host1', 'host2', 'host3']
5
6 services = ['apache_cpuload', 'uptime', 'polling_time', 'proc', 'apache_total_hits', 'apache_total_kbytes']
7 def cacti(teks):
8
9     result = rrdtool.fetch("/opt/cacti/rra/" + teks, "AVERAGE")
10
11     start, end, step = result[0]
12     ds = result[1]
13     rows = result[2]
14
15     AVG = 0
16     TOTAL = 0
17     datavalid = 0
18     for row in rows:
19         val = row[0]
20         if val:
21             TOTAL = TOTAL + val
22             datavalid = datavalid + 1
23
24     AVG = (TOTAL/datavalid) if int(datavalid) != 0 else 0
25     hasil = '%s %s' % (teks, AVG)
26     print(hasil)
27     return(AVG)
28
29
30 registry = CollectorRegistry()
31 file_rrd = os.listdir("/opt/cacti/rra")
32 for service in services:
33     data = Gauge(cacti, '%s' % service, ['hostname'], registry=registry)
34     for host in hosts:
35         file_name = "%(host)s %(service)s"
36         for rrd in file_rrd:
37             if rrd.find(file_name) > -1:
38                 data.labels(host).set(cacti(rrd))
39
40 push_to_gateway('localhost:9091', job='Cacti', registry=registry)
41
42
```

Gambar 5.12 Tampilan Script Python

Agar bisa terkoneksi ke *Pushgateway* dan juga *Prometheus* maka di butuhkan modul-modul untuk mengkoneksikannya. Yang dibutuhkan adalah modul *import rrdtool*, *import os*, *from Prometheus client*, *import CollectorRegistry*, *Gauge*, dan juga *push\_gateway*.



```
abidun@alim: ~/Pushgateway
abidun@alim:~/Pushgateway$ python3 cacti-reading.py
host1_apache_cpuload_480.rrd = 0.03068232290373563
host1_apache_cpuload_637.rrd = 0.030938616303751804
host1_apache_cpuload_626.rrd = 0.030676552596982754
host2_apache_cpuload_652.rrd = 0.03779710892923649
host3_apache_cpuload_644.rrd = 0.18027291777777776
host1_polling_time_641.rrd = 0.26389768959639925
host2_polling_time_655.rrd = 0.6087936899071513
host3_polling_time_648.rrd = 0.3452793367703743
host1_proc_647.rrd = 245.65670995671005
host2_proc_656.rrd = 246.08957169459967
host3_proc_649.rrd = 259.6688888888889
host1_apache_total_kbytes_636.rrd = 0.7827434531978741
host2_apache_total_kbytes_651.rrd = 0.9556435435158303
host3_apache_total_kbytes_643.rrd = 9.39883690708252
abidun@alim:~/Pushgateway$
```

Gambar 5.13 Hasil Run Python

Tampilan gambar diatas merupakan hasil dari *run python cacti-reading.py*. Hasil tersebut akan terbaca oleh *Pushgateway* dan menampilkan hasil *metric* di web *Pushgateway* dan *Prometheus*.



### 5.3.3 Pushgateway

*Pushgateway* mengambil data dari *scripts python cacti-reading.py* dan menampilkan *value* berupa *metric* disetiap *data-data* nya.



Gambar 5.14 Tampilan Web Pushgateway

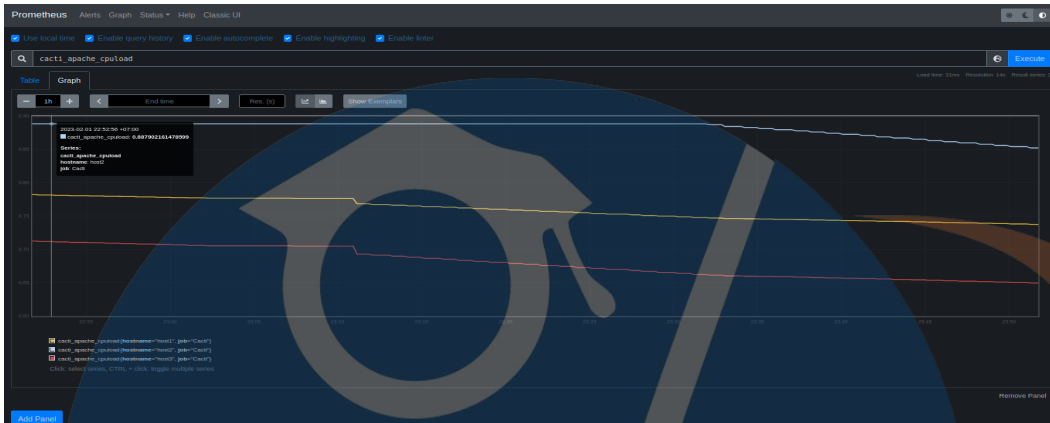
*Pushgateway* membaca *scripts python cacti-reading.py* setelah *run* di terminal dan menampilkan hasilnya di *web Pushgateway*.

### 5.3.4 Prometheus

*Prometheus* menerima hasil *scripts python* dari *Pushgateway*, setelah itu akan menampilkan grafik *Graph* dan juga *value* dari data yang dicari oleh *prometheus*.

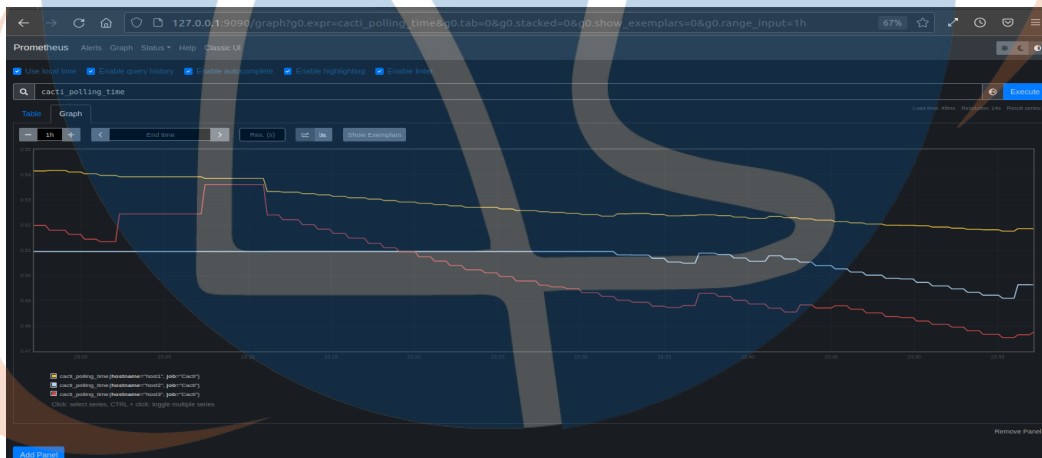
- Menambahkan tampilan grafik *CPU* di *Prometheus*

STT - NF



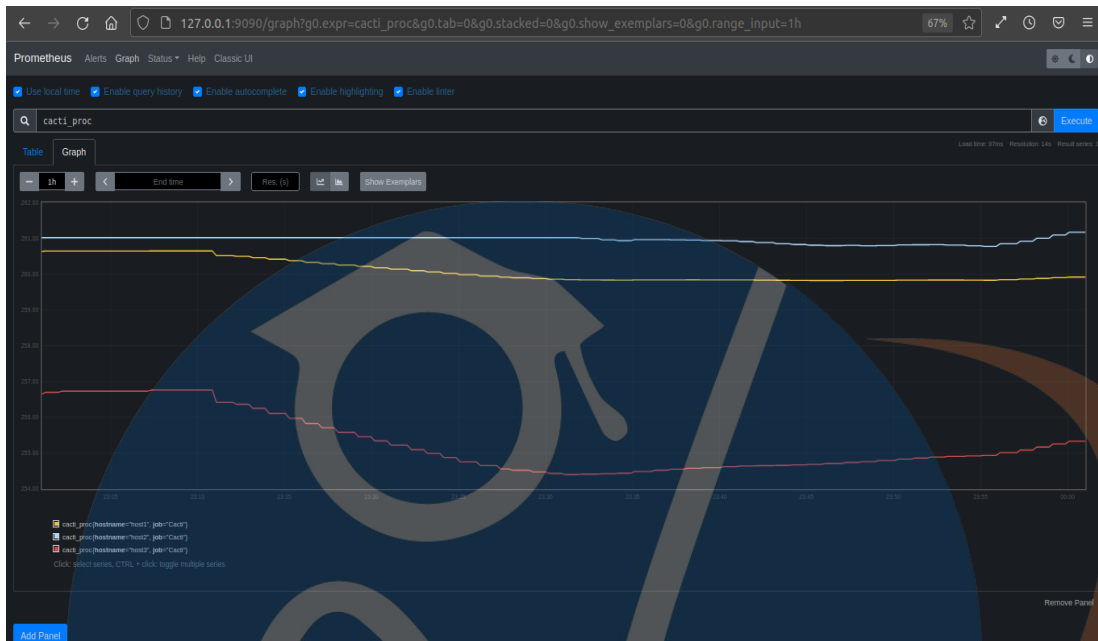
Gambar 5.15 Tampilan Graph CPU Prometheus

- Menambahkan tampilan grafik *pooling time* di Prometheus



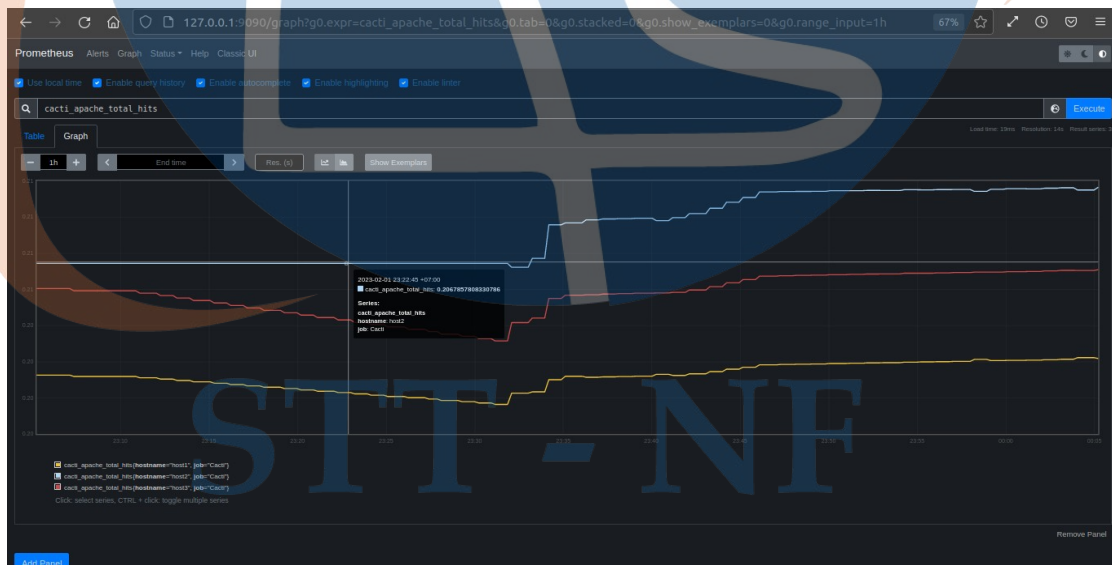
Gambar 5.16 Tampilan Graph Pooling Time Prometheus

- Menambahkan tampilan grafik *processes* di Prometheus



Gambar 5.17 Tampilan Graph Processes Prometheus

- Menambahkan tampilan grafik Bytes/Hit di Prometheus

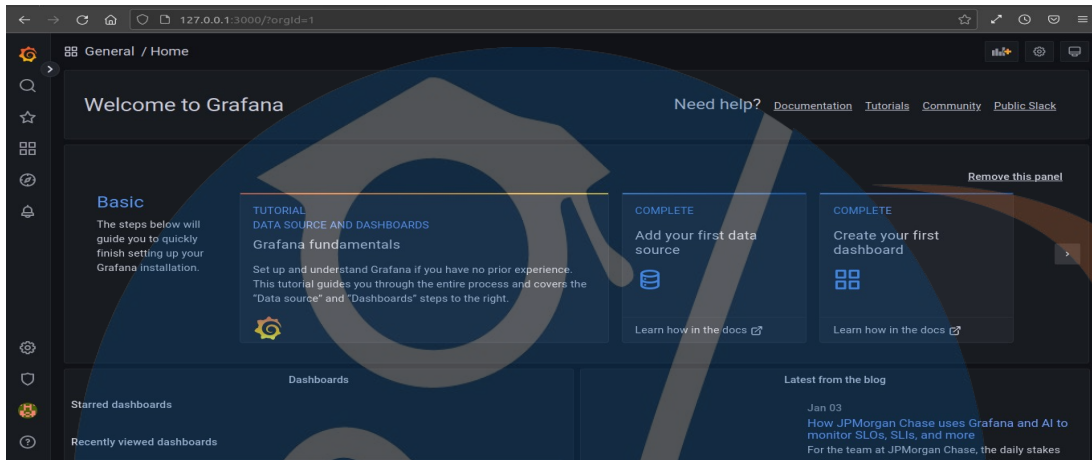


Gambar 5.18 Tampilan Graph Bytes/Hit Prometheus

Prometheus membaca data script python *cacti-reading.py* dari *Pushgateway*, setelah itu *search* untuk mencari data yang ingin di cari, lalu jika sudah menemukannya klik *Execute* untuk mengeksekusinya dan akan menampilkan grafik *graph* dan value data yang sudah di *Execute*.

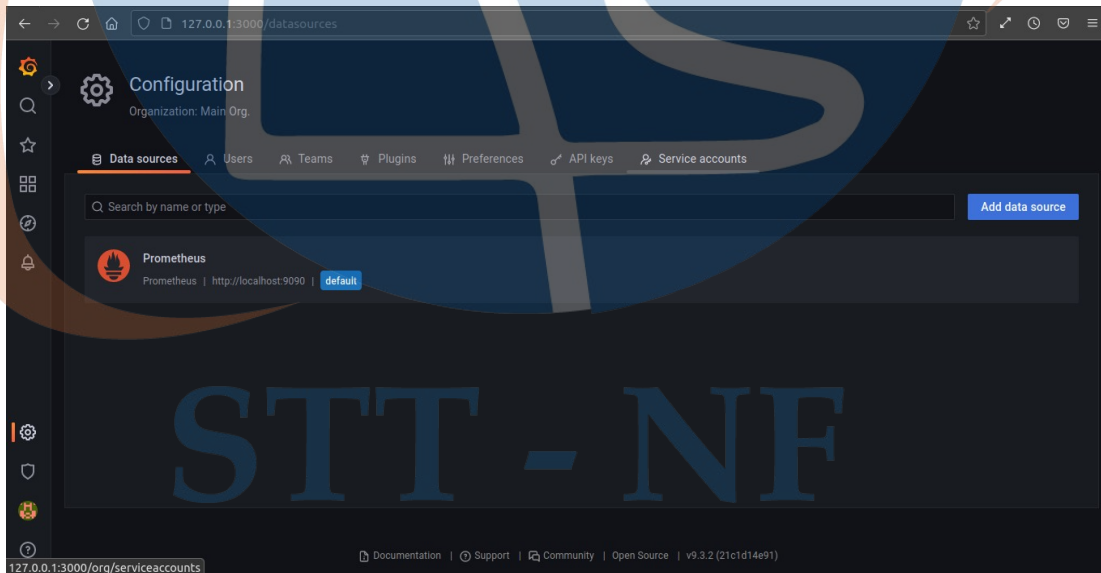
### 5.3.5 Grafana

Untuk bisa menampilkan *dashboard monitoring* maka sebelum itu kita harus *setting Grafana* supaya bisa menampilkan *dashboard* yang nantinya akan ditampilkan.



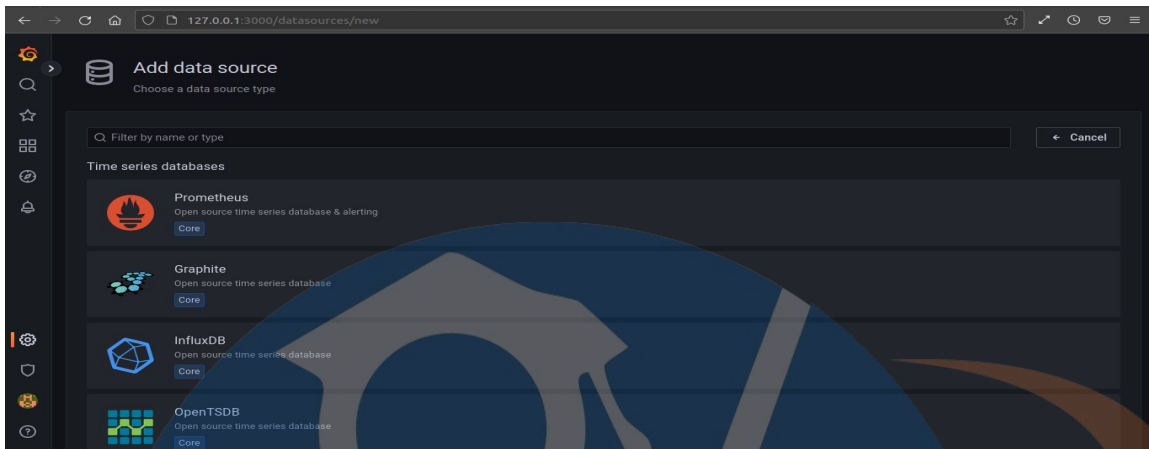
Gambar 5.19 Tampilan Home Grafana

Gambar tampilan diatas merupakan tampilan *Home Grafana*, dengan beberapa *fitur-fitur* untuk mempermudah pekerjaan *Grafana* itu sendiri



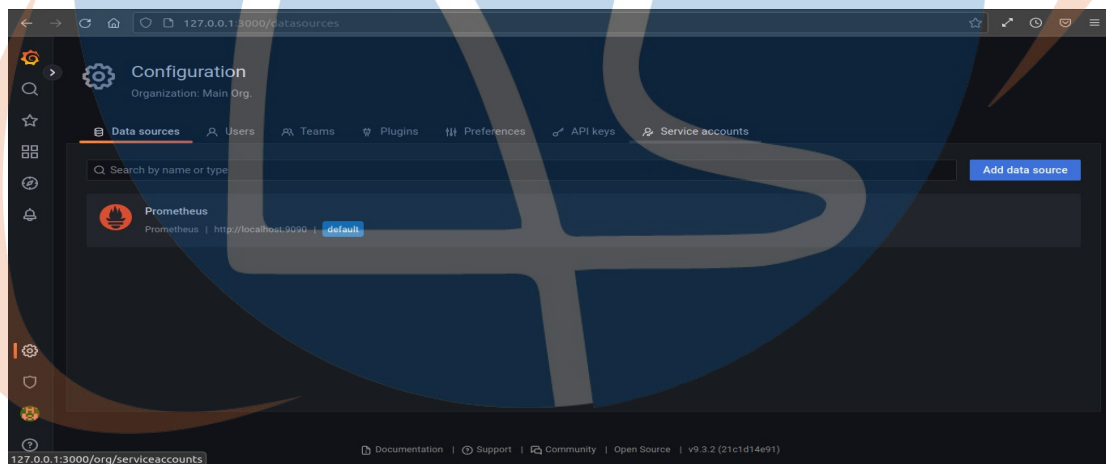
Gambar 5.20 Tampilan Penambahan Data Sources Grafana

Menambahkan *data sources* klik '*configuration*' dengan gambar ikon pengaturan, setelah itu pilih klik *data sources* dan klik '*Add data sources*' untuk menambahkan *data sources* yaitu *Prometheus*.



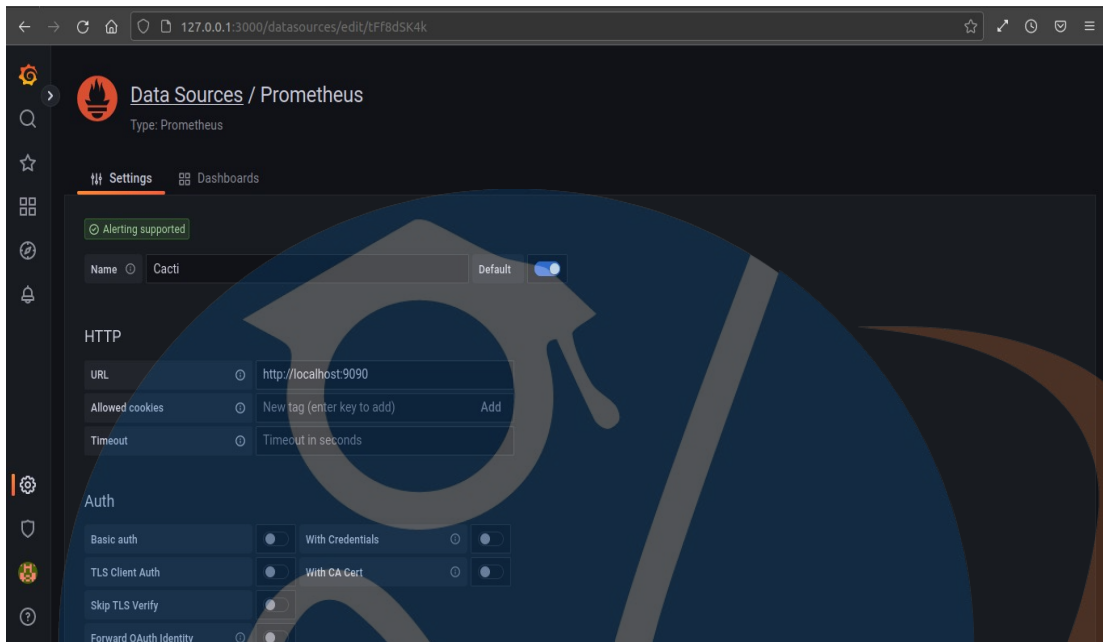
Gambar 5.21 Tampilan Add Data Sources

Tampilan gambar diatas merupakan beberapa dari *data sources database*, pilih klik '*Prometheus*' untuk sebagai *database data sources Grafana*. Setelah memilih '*Prometheus*' maka akan menampilkan gambar sebelumnya dan bertambah *data sources 'Prometheus'* yang sudah di tambahkan sebelumnya.



Gambar 5.22 Tampilan Prometheus Data Source

Tampilan gambar diatas sudah menampilkan ada tambahan *data sources Prometheus*.

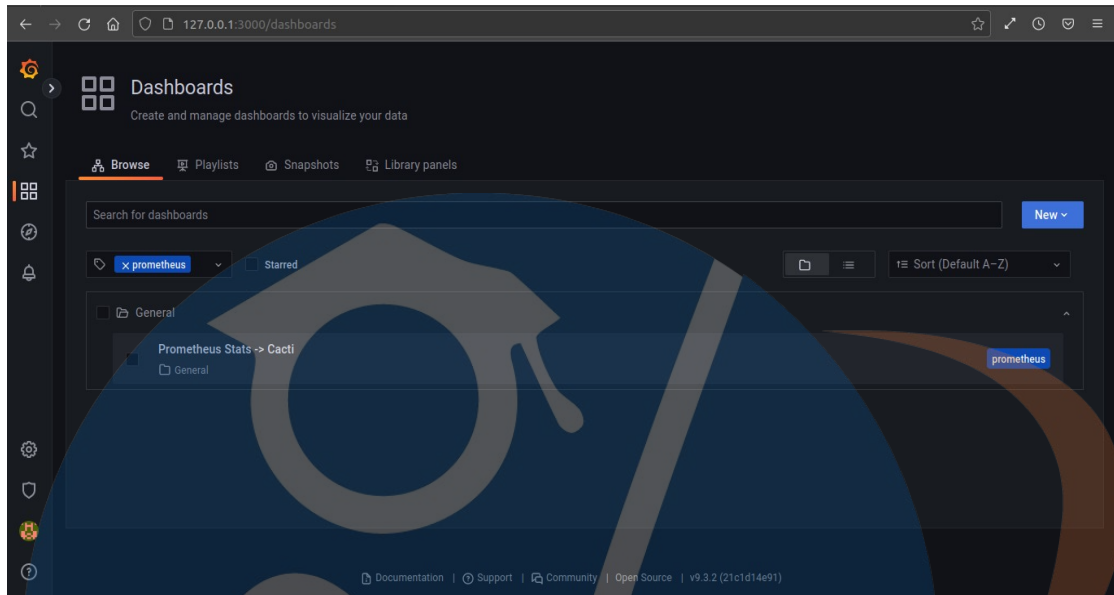


Gambar 5.23 Tampilan Settings Data Source



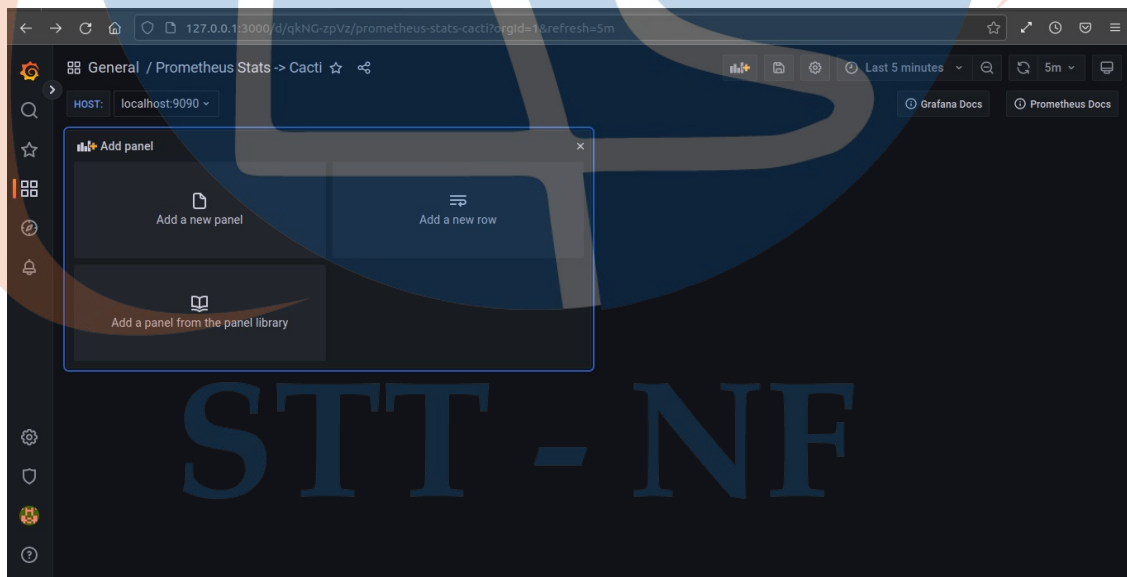
Gambar 5.24 Tampilan Settings Data Sources

Setelah itu klik *Prometheus* dan atur pengaturan *data sources Prometheus* yang dibutuhkan, setelah sudah semua yang dibutuhkan terisi maka klik 'Save & test' untuk menambahkan data *Prometheus* ke dalam *Grafana*.



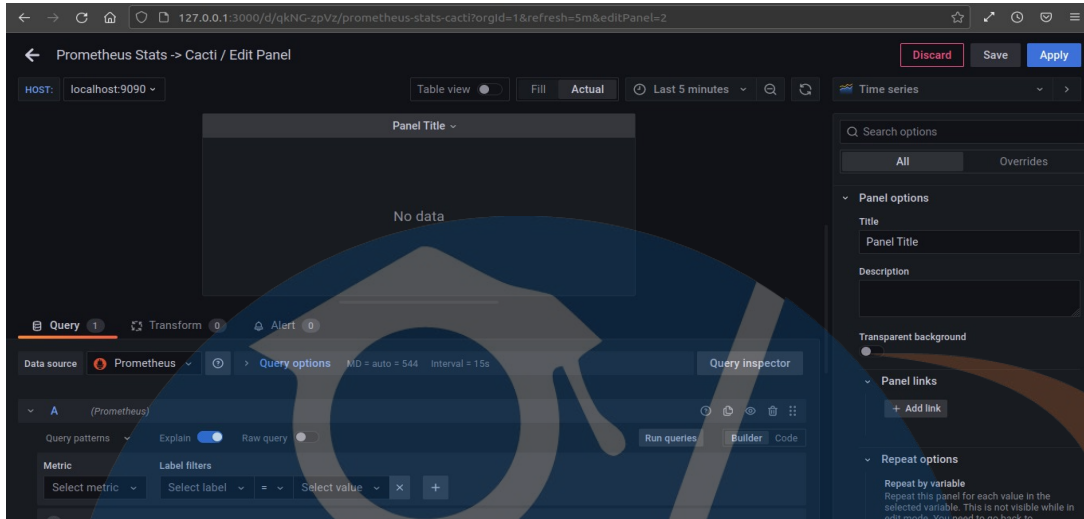
Gambar 5.25 Tampilan Dashboard Prometheus

Klik ikon 'Dashboard' setelah itu pilih klik 'Browse' dan akan menampilkan gambar diatas sudah berhasil *settings Prometheus*.

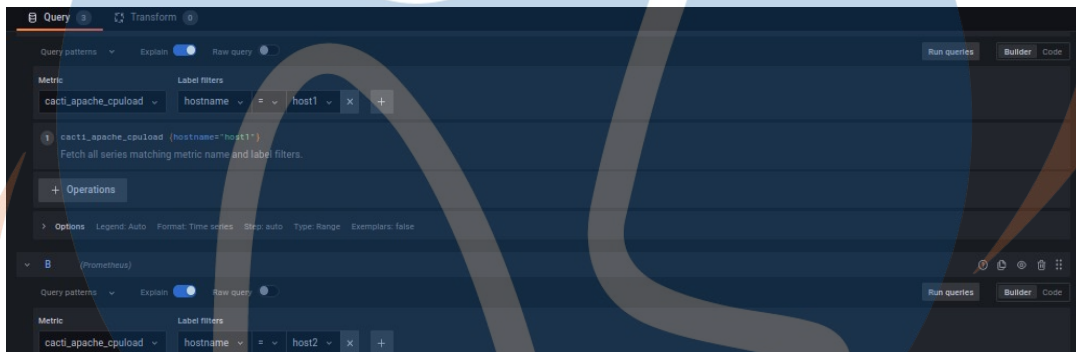


Gambar 5.26 Tampilan Add Panel

Setelah sudah membuat *data sources Prometheus* selanjutnya membuat *dashboard monitoring* dengan cara membuat panel baru. Klik 'Add a new panel' untuk membuat panel baru.

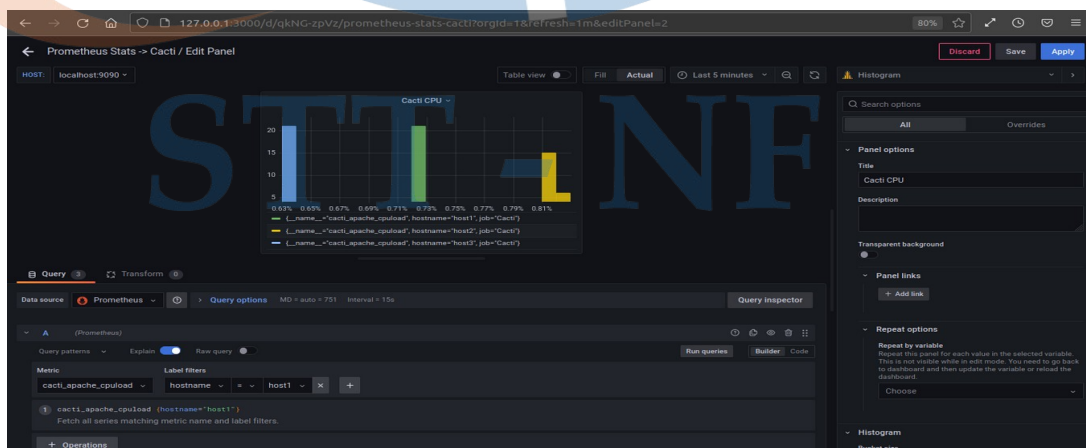


Gambar 5.27 Pembuatan Panel Baru Cacti Cpu



Gambar 5.28 Penambahan Cacti Cpu

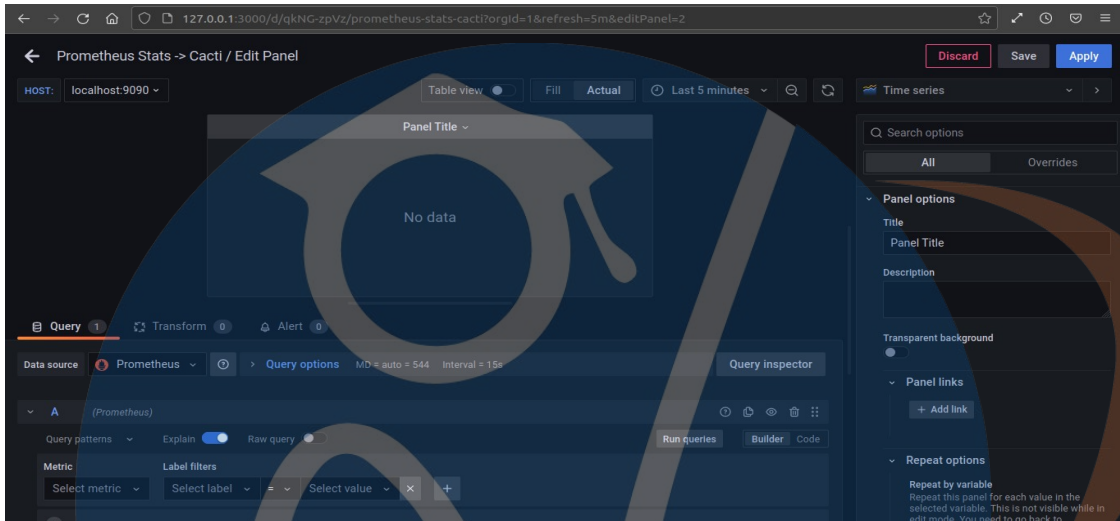
Menambahkan *metric* dan juga *label filters* untuk membuat panel yang akan di *monitoring* yaitu 'host 1, host 2, dan host 3', setelah semuanya sudah sesuai lalu klik 'Run queries'.



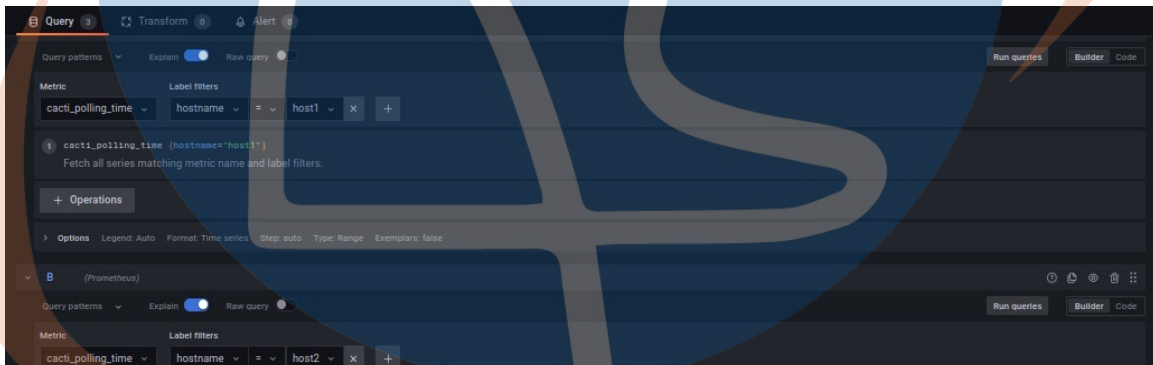
Gambar 5.29 Tampilan Histogram Cacti Cpu



Setelah sudah 'Run queries' maka akan menampilkan grafik *histogram Cacti Cpu* dan bisa mengetahui berapa *value, times series* yang *Cpu host 'Cacti Cpu'*. Klik 'Apply' untuk menampilkan *dashboard Cacti Cpu* di *Grafana*.

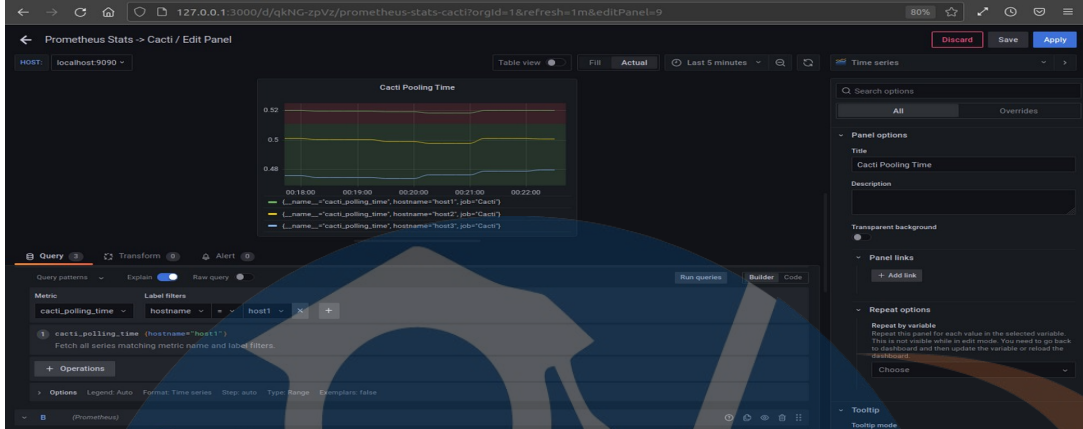


Gambar 5.30 Pembuatn Panel baru Cacti Pooling Time



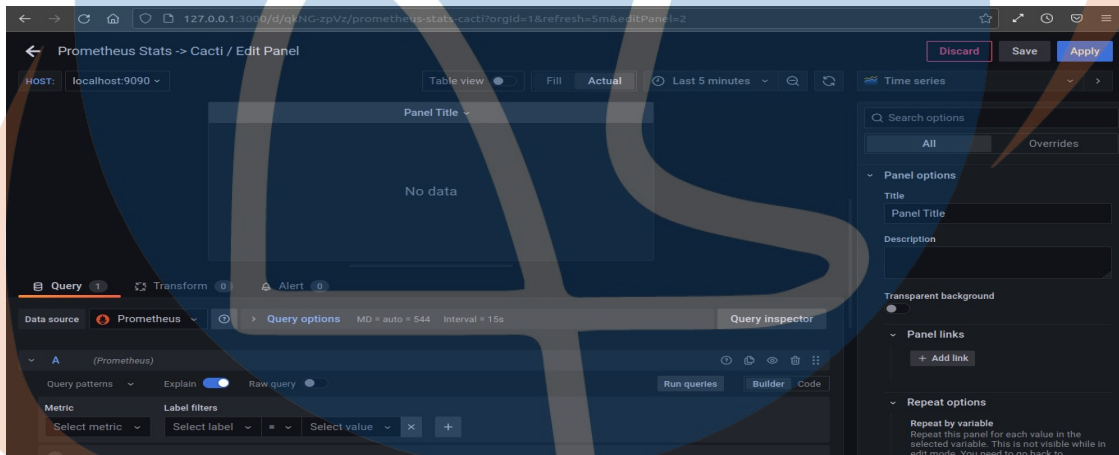
Gambar 5.31 Penambahan Cacti Pooling Time

Menambahkan *metric* dan juga *label filters* untuk membuat panel yang akan di *monitoring* yaitu '*cacti\_polling\_time*' *host 1, host 2, dan host 3*, setelah semuanya sudah sesuai lalu klik '*Run queries*'.



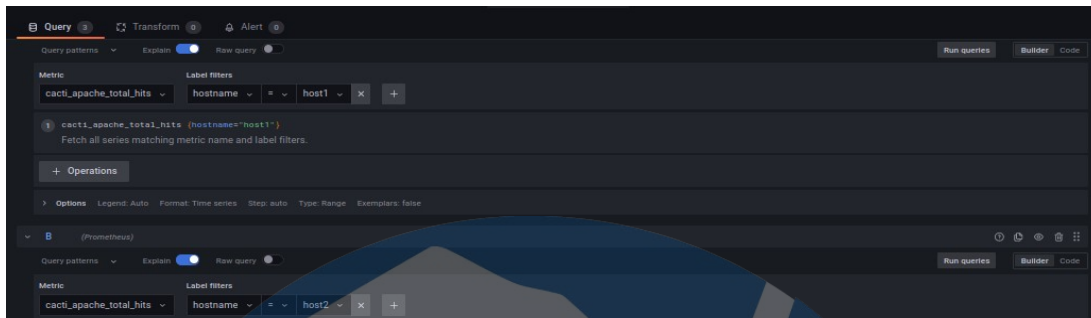
Gambar 5.32 Tampilan Grafik Cacti Pooling Time

Setelah sudah 'Run queries' maka akan menampilkan grafik Cacti Pooling Time dan bisa mengetahui berapa value, times series yang Pooling Time host 'Cacti Pooling Time'. Klik 'Apply' untuk menampilkan dashboard Cacti Pooling Time di Grafana.



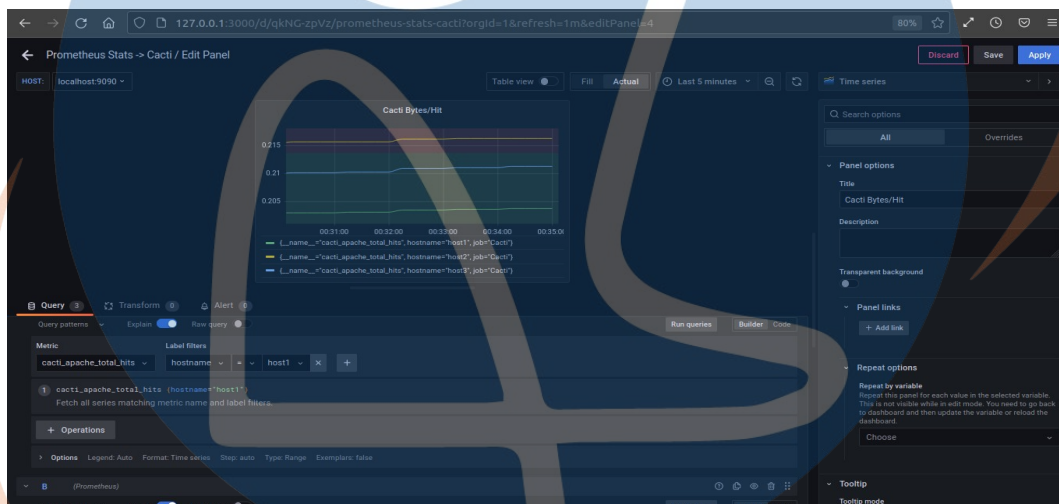
Gambar 5.33 Pembuatan Panel baru Cacti Bytes/Hit

STT - NF



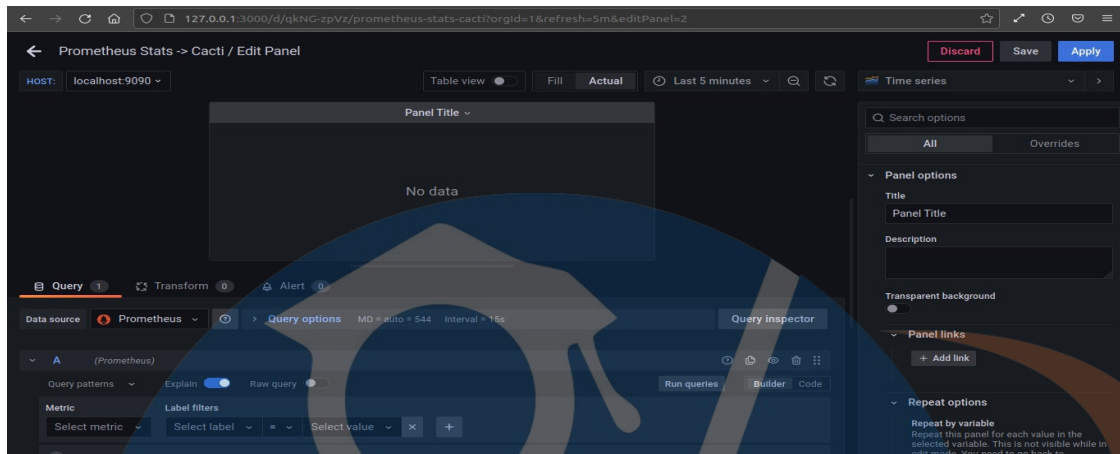
Gambar 5.34 Penambahan Cacti Bytes/Hit

Menambahkan *metric* dan juga *label filters* untuk membuat panel yang akan di *monitoring* yaitu 'cacti\_apache\_total\_hits' host 1, host 2, dan host 3, setelah semuanya sudah sesuai lalu klik 'Run queries'

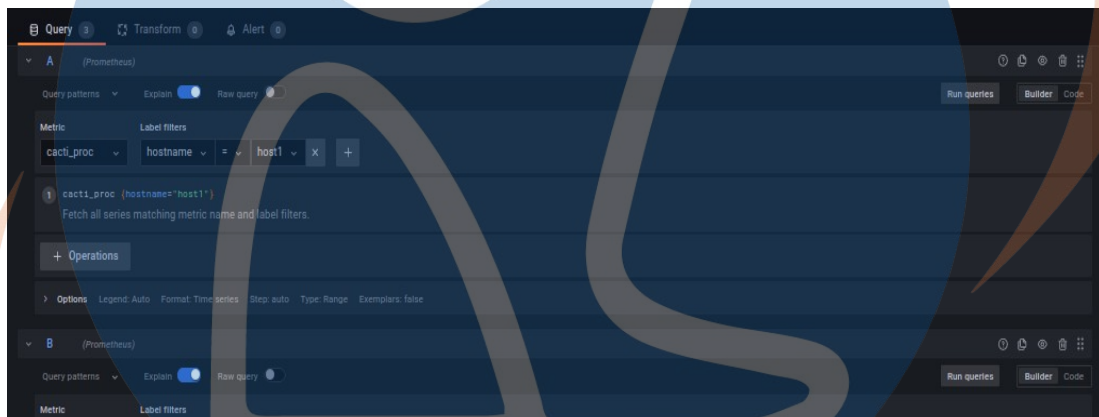


Gambar 5.35 Tampilan Grafik Cacti Bytes/Hit

STT - NF



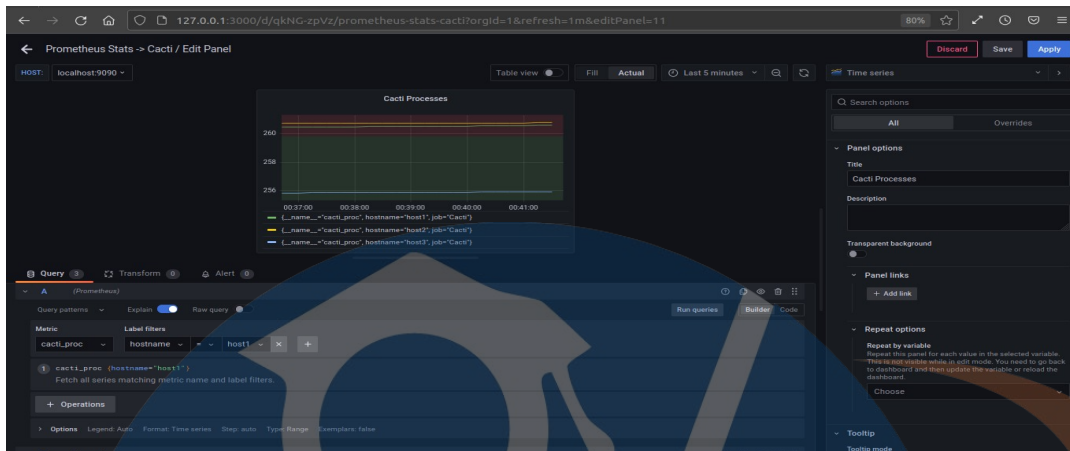
Gambar 5.36 Penambahan Panel Cacti Processes



Gambar 5.37 Penambahan Cacti Processes

Menambahkan *metric* dan juga *label filters* untuk membuat panel yang akan di *monitoring* yaitu 'cacti\_proc' host 1, host 2, dan host 3, setelah semuanya sudah sesuai lalu klik 'Run queries'.

STT - NF



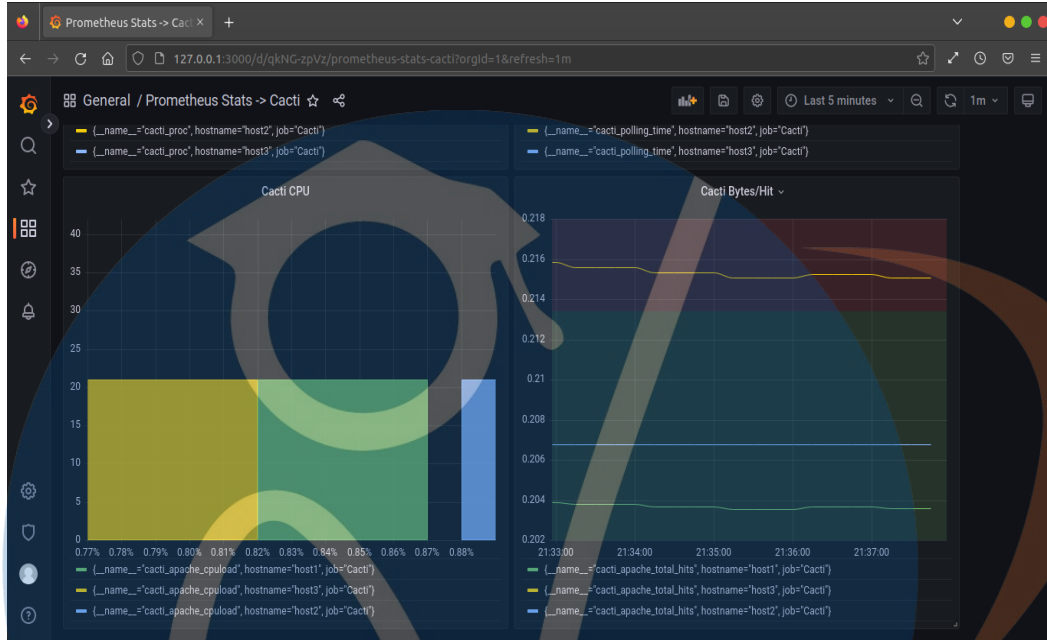
Gambar 5.38 Tampilan Grafik Cacti Processes

Setelah sudah 'Run queries' maka akan menampilkan grafik Cacti Processes dan bisa mengetahui berapa value processes atau yang sedang host jalankan di 'host 1, host 2, dan host 3'. Klik 'Apply' untuk menampilkan dashboard processes di Grafana.

Berikut adalah hasil dari dashboard sistem dan jaringan berbasis Cacti menggunakan Grafana yang melalui proses pembuatan scripts python cacti-reading.py di push ke dalam Pushgateway dan Prometheus.



Gambar 5.39 Hasil Dashboard Monitoring Grafana



Gambar 5.40 Hasil Dashboard Monitoring Grafana

Berikut adalah kesimpulan dari pengujian implementasi *dashboard* untuk *monitoring* sistem dan jaringan berbasis *Cacti* menggunakan *Grafana* yang telah dilakukan. Pengujian efektivitas *dashboard Grafana* dalam perancangan dan implementasi *dashboard* untuk *monitoring* jaringan yang diuji berhasil dilakukan uji coba.

STT - NF

Tabel 5.1 Hasil Pengujian Efektifitas

No	Pengujian	Ekspektasi	Hasil
1	Menambahkan Device baru di <i>Cacti</i>	<i>Device</i> bertambah di <i>Cacti</i>	Berhasil
2	<i>Device Data sources</i> di <i>Cacti</i> bertambah	<i>Data sources Device</i> bertambah	Berhasil
3	Membuat <i>Script python cacti-reading.py</i> untuk mengambil data <i>Cacti</i> dan mengepush ke <i>Pushgateway, Prometheus, Grafana</i>	<i>Script python cacti-reading.py</i> terbuat dan terkoneksi ke <i>Pushgateway, Prometheus, Grafana</i>	Berhasil
4	Membuat dan menampilkan panel <i>Dashboard</i> di <i>Grafana</i>	<i>Dashboard</i> berhasil di buat dan di tampilkan di <i>Grafana</i>	Berhasil

Tabel 5.2 Hasil Pembacaan Data dari *Cacti* untuk disimpan ke *Prometheus*

No	Monitoring	Ekspektasi	Hasil
1	Host 1 127.0.0.1	Data dari <i>Cacti</i> , <b>Cpu</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
2	Host 1 127.0.0.1	Data dari <i>Cacti</i> , <b>Pooling time</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
3	Host 1 127.0.0.1	Data dari <i>Cacti</i> , <b>Processes</b> berhasil ditampilkan oleh	Tampil

		<i>Prometheus</i>	
4	Host 1 127.0.0.1	Data dari <i>Cacti</i> , <b>Bytes/Hit</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
5	Host 2 192.168.44.19	Data dari <i>Cacti</i> , <b>Cpu</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
6	Host 2 192.168.44.19	Data dari <i>Cacti</i> , <b>Pooling time</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
7	Host 2 192.168.44.19	Data dari <i>Cacti</i> , <b>Processes</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
8	Host 2 192.168.44.19	Data dari <i>Cacti</i> , <b>Bytes/Hit</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
9	Host 3 192.168.43.20	Data dari <i>Cacti</i> , <b>Cpu</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
10	Host 3 192.168.43.20	Data dari <i>Cacti</i> , <b>Pooling time</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil
11	Host 3 192.168.43.20	Data dari <i>Cacti</i> , <b>Processes</b> berhasil ditampilkan oleh	Tampil



		<i>Prometheus</i>	
12	Host 3 192.168.43.20	Data dari <i>Cacti</i> , <b>Bytes/Hit</b> berhasil ditampilkan oleh <i>Prometheus</i>	Tampil

Tabel 5.3 Hasil Rancangan Pengujian Visualisasi Grafana

No	Pengujian	Deskripsi	Jenis Visualisasi	Hasil Uji	Presentase
1	Cpu	Menampilkan visualisasi Cpu pada <i>Grafana</i>	Histogram	Berhasil di Tampilkan	100%
2	Pooling Time	Menampilkan visualisasi Pooling Time pada <i>Grafana</i>	Time Series dan Grafik	Berhasil di Tampilkan	100%
3	Processes	Menampilkan visualisasi pada <i>Grafana</i>	Time Series dan Grafik	Berhasil di Tampilkan	100%
4	Bytes/Hit	Menampilkan visualisasi pada <i>Grafana</i>	Time Series dan Grafik	Berhasil di Tampilkan	100%