Power Measurement on Chameleon Cloud

Author: Simone Garau

In this tutorial we'll discuss how it's possible to track energy consumption of our instances on Chameleon. We'll use tools like Scaphandre and Prometheus.
Scaphandre is a tool designed to collect energy consumption metrics on linux systems. It uses sensors such as RAPL to monitor and measure the power consumption of CPUs and other hardware components.
Prometheus is an open-source monitoring system; we will use it to collect and store energy consumption metrics.
It's important to note that this measurement can be applied only to bare-metal instances i.e instances created in CHI sites and not in KVM.

Instances creation

First of all we need to create a new instance, but before, in CHI sites we have to **reserve a Lease**.
1 - Open the website https://chi.tacc.chameleoncloud.org/project/
2 - On the left sidebar go down and click on Reservations and then Leases;
3 - On the right side select the Create Lease button;
4 - Insert the name of the lease and the number of days and then click Next;
5 - In the Hosts tab check Reserve hosts and increase the maximum number of hosts to 3 (in our case we'll use 3 instances). Then in Resource properties select node type = compute_haswell_ib;
6 - In the Networks tab check Reserve Floating IPs and insert 1. Then click on the Create button.

Now we have a lease and we can **create the instances**. We'll create 3 instances where one of them will be the main instance and the others will send all the energy consumption metrics. The main instance collects the metrics and the user can visualize them on the browser.
1 - Click Compute in the left menu and then Instances;
2 - Now click the button Launch Instance on the right;
3 - Insert name and select from the dropdown menu the Reservation created before. Increase the counter to 3 (in this way we'll create 3 identical instances at once) and then click next;
4 - Select an image like CC-Ubuntu20.04;
5 - In Networks tab select sharednet1;
6 - In Key Pair we can create or import key pair (you can already have a key pair allocated);
7 - Down, click on show advanced button, select Configuration and insert the script below:

```
#!/bin/bash
apt-get update
```

```
apt install freeipmi-tools -y
apt-get install "linux-tools-$(uname -r)" -y
modprobe intel_rapl_msr
apt-get install pkg-config
apt-get install cargo -y
apt-get install libssl-dev -y
apt-get install ca-certificates curl
```

8 - Then click Launch Instance.

Before we start to work on the instances, we need to **associate an IP** to our new instances. Here are the steps to follow for each instance:
1 - On the left menu, click Network and then Floating IPs;
2 - Choose an available IP Address from the table and click on Associate;
3 - In the new tab select the name of the instance created before.
At this moment we can connect to the new instance using SSH and the address just associated (ssh cc@<floating_ip>).

Instances configuration

Now we have 3 identical instances: you can choose one of them and it'll be the main instance. First in each instance we have to install scaphandre. Type:

```
cargo install scaphandre
```

Then:

```
echo 'export PATH=/home/cc/.cargo/bin:$PATH' >> ~/.bashrc
source ~/.bashrc
```

Main Instance

In the main instance, run:

```
sudo apt-get update
```

Then, to install prometheus that provide to listen to the metrics, run:

```
sudo apt-get install -y prometheus
```

Now we have to tell Prometheus which instances it should listen to. Go to */etc/prometheus* and edit the prometheus.yml file adding the instances. Use nano prometheus.yml and go down until you reach the "scrape_configs:" section; at the bottom of the file add these lines:

- job_name: 'scaphandre_instance_1'

static_configs:
- targets: ['10.52.2.137:8080'] # Change this

- job_name: 'scaphandre_instance_2'
static_configs:
- targets: ['10.52.3.72:8080'] # Change this

I gave job names as scaphandre_instance_1 and scaphandre_instance_2, relatively referring to vm-2 and vm-3. **Remember to replace the target IPs with yours.**
By default, the metrics are exposed on port 8080.
Now type `sudo nano /lib/systemd/system/prometheus.service` and edit the line “ExecStart” with this:

ExecStart=/usr/bin/prometheus --web.enable-admin-api --web.listen-address=0.0.0.0:9090

To check that everything is working we can run the following commands:

```
sudo systemctl daemon-reload
sudo systemctl restart prometheus
sudo systemctl status prometheus
```

Secondary instances

In each instance that we want to monitorate, first we check if scaphandre is correctly installed (you can just type scaphandre and see if you got instructions on how to use it). In case scaphandre is not installed you can type:

```
cargo install scaphandre
```

After that we run:

```
scaphandre prometheus
```

In this way, the instance is exposing metrics and Prometheus running on the main vm will scrape them.

Firewall rules

By default Chameleon Ubuntu base images come with baked-in firewall rules which severely restrict connection over the public internet. This firewall utility is called `firewalld` and by default exposes only port 22 for SSH connections. In order to make the services work correctly, we need to apply the following changes.

Prometheus

To verify the Prometheus web-UI is currently working we can reach it with `curl localhost:9090` and you should receive the following response:

```
<a href="/graph">Found</a>.
```

But, if you try to visit <MainVM_Floating_IP>:9090 you'll end up with "Connection timed out" for the reason explained above. To make the web-UI reachable using your floating ip on your browser you have to type:

```
sudo firewall-cmd --zone=public --add-port=9090/tcp
```

Scaphandre

Regarding the two instances running Scaphandre, we don't need to expose it on the public network, we simply need to make them reachable on the default *sharednet* provided by Chameleon. To do this, go to your Chameleon "Instances" tab, look at what subnet your machines are running (e.g in my case the subnet is 10.52.0.0/16).

vm-3	CC-Ubuntu20.04	10.52.3.72,
vm-2	CC-Ubuntu20.04	10.52.2.137
vm-1	CC-Ubuntu20.04	10.52.3.13,

then type the following command to make scaphandre metrics available.
`sudo firewall-cmd --zone=trusted --add-source=10.52.0.0/16` (replace 10.52.0.0/16 with your subnet).

Persistence & Security

Please note that these actions about the firewall are not permanent across reboots, so you can use:

```
sudo firewall-cmd -runtime-to-permanent
sudo firewall-cmd --reload
```

Additionally, Prometheus by default does not have an authentication mechanism enabled, so making it accessible from the public can expose information of your cluster to everyone. For our demo it is not needed but If you plan to use it for real projects it's highly recommended to secure it, for instance using this guide: Basic auth | Prometheus.

Troubleshooting

If a permission denied error occurs you can follow this guide: Troubleshooting - Scaphandre documentation.
Also, if you get this error: `thread 'main' panicked at /home/cc/.cargo/registry/src/index.crates.io-6f17d22bba15001f/hyper-0.14.30/src/server/server.rs:81:13:`

```
error binding to 0.0.0.0:8080: error creating server listener:
Address already in use (os error 98)
note: run with `RUST_BACKTRACE=1` environment variable to display
a backtrace
```

it's because there's another scaphandre process running, so you have to kill it with:

```
sudo lsof -i:8080
kill -9 PID_NUMBER
```

Grafana

For this part I'll use the files provided in the Scaphandre git repo. These are originally made for docker but can be used in our testbed.
Follow these steps for the installation process:

```
sudo apt-get install -y apt-transport-https
software-properties-common wget
sudo mkdir -p /etc/apt/keyrings/
wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor |
sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null
echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg]
https://apt.grafana.com stable main" | sudo tee -a
/etc/apt/sources.list.d/grafana.list
sudo apt-get update
sudo apt-get install grafana
sudo systemctl daemon-reload
sudo systemctl start grafana-server
sudo systemctl status grafana-server
```

Now we'll make some customizations in order to configure the datasource and a sample dashboard. Execute `sudo -s` for root privileges. Then:

- Go to `/etc/grafana/provisioning/datasources`
- `rm sample.yaml`
- Create a file called datasource.yml and paste this content:

```
# config file version
apiVersion: 1


datasources:
  # <string, required> name of the datasource. Required
- name: Prometheus-scaph
  # <string, required> datasource type. Required
  type: prometheus
  # <string, required> access mode. direct or proxy. Required
  access: proxy
  # <int> org id. will default to orgId 1 if not specified
  orgId: 1
  # <string> url
  url: http://localhost:9090
```

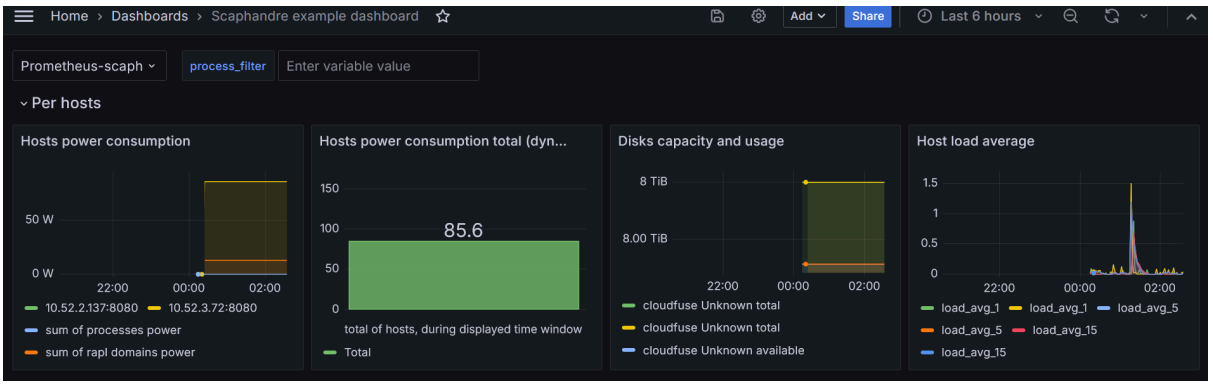
isDefault: true
version: 1
 # <bool> allow users to edit datasources from the UI.
editable: true

- `cd ../dashboards`
- Create dashboard.yml and paste this content:
  apiVersion: 1

  providers:
  # <string> an unique provider name. Required
  - name: 'Scaphandre examples'
  # <int> Org id. Default to 1
  orgId: 1
  # <string> name of the dashboard folder.
  folder: 'scaphandre'
  # <string> provider type. Default to 'file'
  type: file
  # <bool> disable dashboard deletion
  disableDeletion: true
  # <bool> allow updating provisioned dashboards from the UI
  allowUiUpdates: false
  options:
  # <string, required> path to dashboard files on disk. Required when using the 'file' type
  path: /var/lib/grafana/dashboards
  # <bool> use folder names from filesystem to create folders in Grafana
  foldersFromFilesStructure: true

- `cd /var/lib/grafana`
- `mkdir dashboards && cd dashboards`
- `curl -O https://raw.githubusercontent.com/hubblo-org/scaphandre/main/docker-compose/dashboards/sample-dashboard.json`
- `systemctl restart grafana-server`

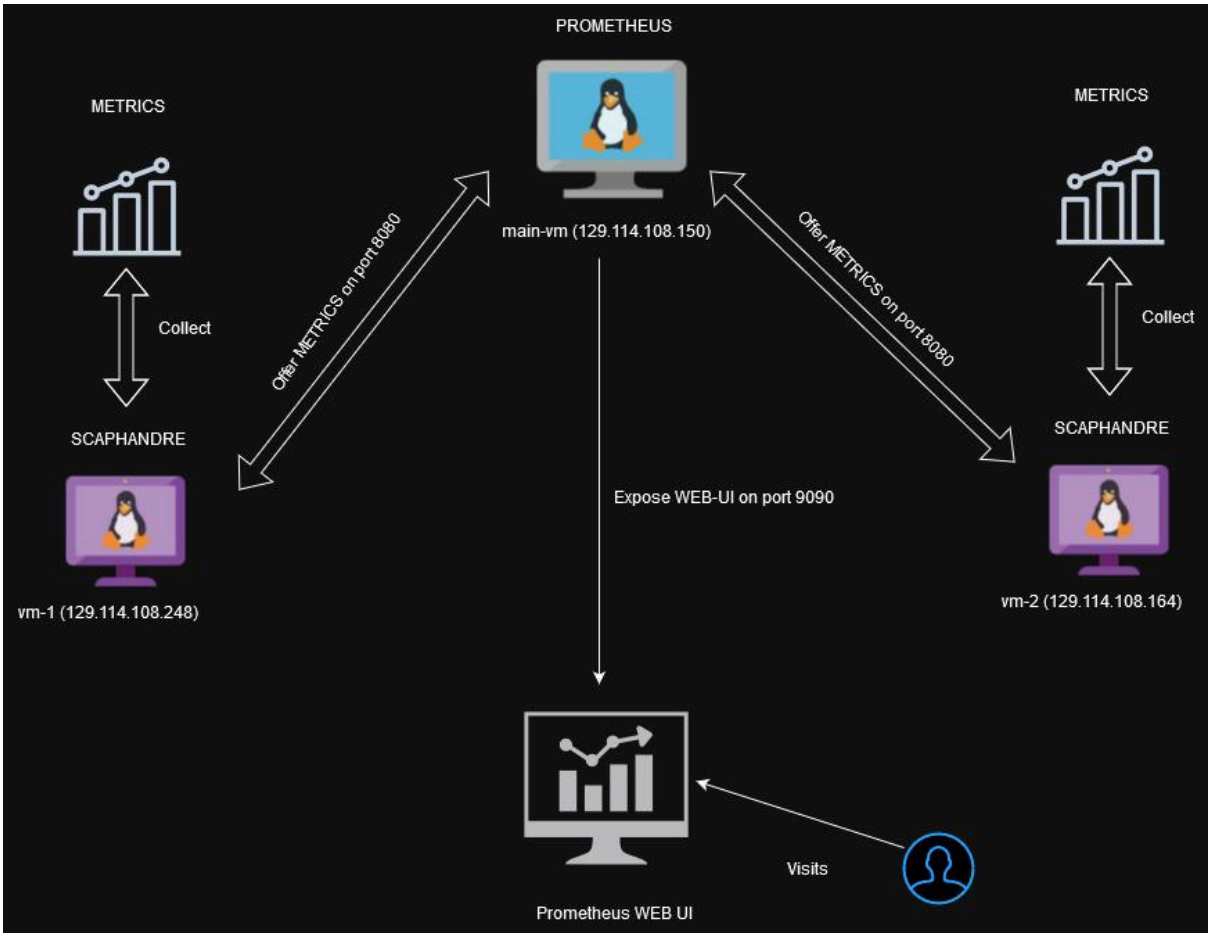
Now go to <Floating_IP_MainInstance>:3000. Follow the steps for the login. Then:
1 - Open Home
2 - Dashboards
3 - You should have a dashboard called *Scaphandre example dashboard*



4 - Enjoy your dashboard.

Summarizing

We have 2 instances of which we want to control the energy consumption. In these 2 instances we've installed Scaphandre that makes metrics available to an http endpoint at port 8080. Also, we have a main instance where Prometheus is installed that is configured to do scraping of metrics from the other two instances running Scaphandre. Finally, the Prometheus web-UI can be accessed to view the metrics obtained from our 2 instances and Grafana can be used for a better visualization of data.



Now, to test if Prometheus and Grafana are working correctly, open your browser and go to:

<Floating_IP_main_instance>:9090

and, for Grafana:

<Floating_IP_main_instance>:3000

Simone Garau - 20034352@studenti.uniupo.it