DevOps

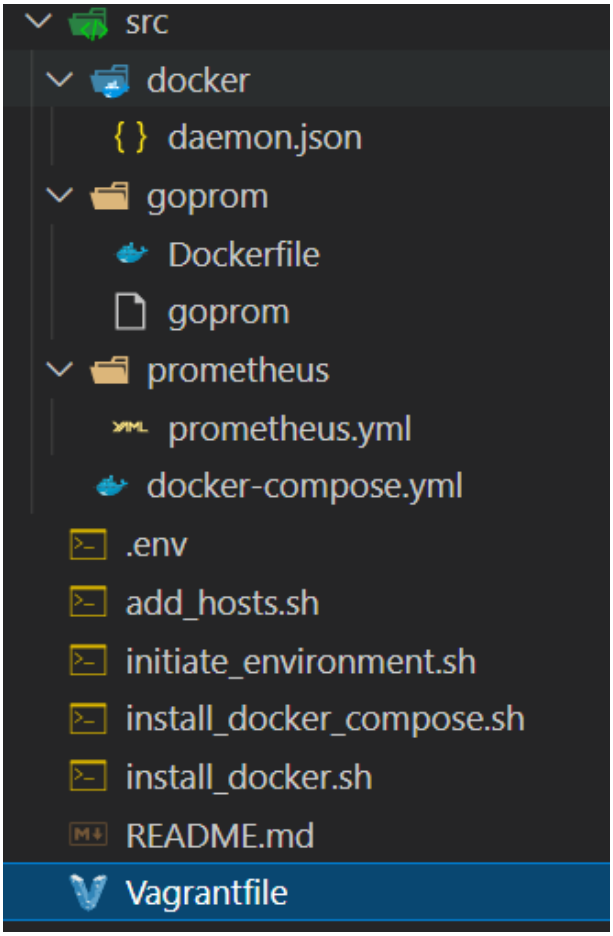
Containerization, CI/CD & Monitoring

February 2022

Monitoring with Prometheus and Grafana

Home Work

Stefan Veselinov



Assignment

You are expected to create the following

- A setup with one virtual machine with **Docker** installed just like on the practice
- On the machine(s) run the following
    - **Prometheus** as container. One instance
    - **Grafana** as container. One instance
    - The application (**goprom**) used during the practice. Two instances
- In terms of measurement, do the following
    - Make **Docker** to provide metrics, which to be consumed by **Prometheus**. This should result in one job. *For this one, you should research **Docker** documentation*
    - Capture the metrics of the two application instances. This should result in one job with two targets
- In terms of visualization, create a simple dashboard that has
    - A panel which shows how many containers are on the host (in all states)
    - A panel which shows how many jobs are processed by each **goprom** application (all result types)

Preparation

- Host Machine - Windows 10 Pro 21H2 19044.1526
- Creating Project structure
    - src/ folder that is going to be shared throughout vm's (contains: docker daemon file for exposing metrics, goprom app, Prometheus config file, docker-compose file for container orchestration)
    - root – Vagrant file and all .sh scripts for vm's setup
- Vagrantfile – create one vm.

```
Vagrant.configure(2) do |config|

  config.env.enable
  config.ssh.insert_key = false

  config.vm.define ENV['NODE_VM_HOST_NAME'] do |node|
    node.vm.box=ENV['NODE_VAGRANT_BOX_NAME']
    node.vm.hostname = ENV['NODE_VM_HOST_NAME']
    node.vm.provider :virtualbox do |vb|
      vb.memory = ENV['NODE_VB_MEMORY']
      vb.cpus = ENV['NODE_VB_CPUS']
      vb.customize ["modifyvm", :id, "--usb", "off"]
      vb.customize ["modifyvm", :id, "--usbehci", "off"]
    end
```

```
    node.vm.network "private_network", ip: ENV['NODE_VM_IP']
    node.vm.provision "shell", path: ENV['NODE_VM_ADD_HOSTS_SCRIPT']
    node.vm.provision "shell", path: ENV['NODE_VM_DOCKER_SCRIPT']
    node.vm.provision "shell", path: ENV['NODE_VM_DOCKER_COMPOSE_SCRIPT']
    node.vm.provision "shell", path: ENV['NODE_VM_ENVIRONMENT_SCRIPT']
  end
end
```

- Vagrant .env file

```
export NODE_VM_DEFINE_NAME=node
export NODE_VAGRANT_BOX_NAME=shekeriev/debian-11
export NODE_VB_MEMORY=2048
export NODE_VB_CPUS=1
export NODE_VM_HOST_NAME=node.do1.homework
export NODE_VM_IP=192.168.99.100
export NODE_VM_ADD_HOSTS_SCRIPT=add_hosts.sh
export NODE_VM_DOCKER_SCRIPT=install_docker.sh
export NODE_VM_DOCKER_COMPOSE_SCRIPT=install_docker_compose.sh
export NODE_VM_ENVIRONMENT_SCRIPT=initiate_environment.sh

export ADD_HOSTS_SCRIPT=add-hosts.sh
```

- NODE_VM_ADD_HOSTS_SCRIPT

```
echo "192.168.99.100 node.do1.homework node" >> /etc/hosts
```

- NODE_VM_DOCKER_SCRIPT

```
echo "* Update repositories and install common packages"
apt-get update
apt-get install -y ca-certificates curl gnupg lsb-release tree

echo "* Add Docker repository key"
curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg

echo "* Add Docker repository"
echo "deb [arch=$(dpkg --print-architecture) signed-
by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/debian $(lsb_release -cs) stable" | tee
/etc/apt/sources.list.d/docker.list > /dev/null

echo "* Install Docker"
apt-get update
apt-get install -y docker-ce docker-ce-cli containerd.io

echo "* Adjust group membership"
```

```
usermod -aG docker vagrant
```

- NODE_VM_DOCKER_COMPOSE_SCRIPT

```
echo "* Adding Docker-Compose"
sudo curl -L
"https://github.com/docker/compose/releases/download/1.29.2/docker-compose-
$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

- NODE_VM_ENVIRONMENT_SCRIPT (copy docker daemon.json file, restart service and run docker-compose file)

```
echo "** Configure Docker"
cp /vagrant/src/docker/daemon.json /etc/docker/daemon.json

echo "** Restart Service"
systemctl restart docker

echo "** Run Docker Compose"
cd /vagrant/src && docker-compose up -d –build
```

- Docker daemon file – expose docker metrics

```
{
       "metrics-addr": "192.168.99.100:7070",
       "experimental": true
}
```

- Docker-compose file

```
version: "3"
services:
  prometheus:
    image: prom/prometheus
    container_name: prometheus
    networks:
      - goprom
    volumes:
      - ./prometheus/prometheus.yml:/etc/prometheus/prometheus.yml
    ports:
      - "9090:9090"
  grafana:
    image: grafana/grafana
    container_name: grafana
    ports:
      - 3000:3000
    networks:
      - goprom
    depends_on:
```

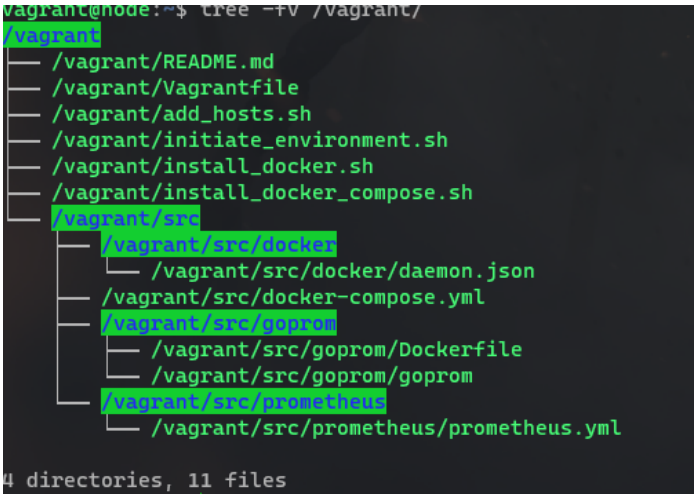
```
      - prometheus
  goprom_node1:
    container_name: goprom_node1
    build:
      context: ./goprom/
      dockerfile: Dockerfile
    networks:
      - goprom
    ports:
      - 8090:8080
    depends_on:
      - prometheus
      - grafana
  goprom_node2:
    container_name: goprom_node2
    build:
      context: ./goprom/
      dockerfile: Dockerfile
    networks:
      - goprom
    ports:
      - 8099:8080
    depends_on:
      - prometheus
      - grafana
networks:
  goprom:
```

- Prometheus configuration

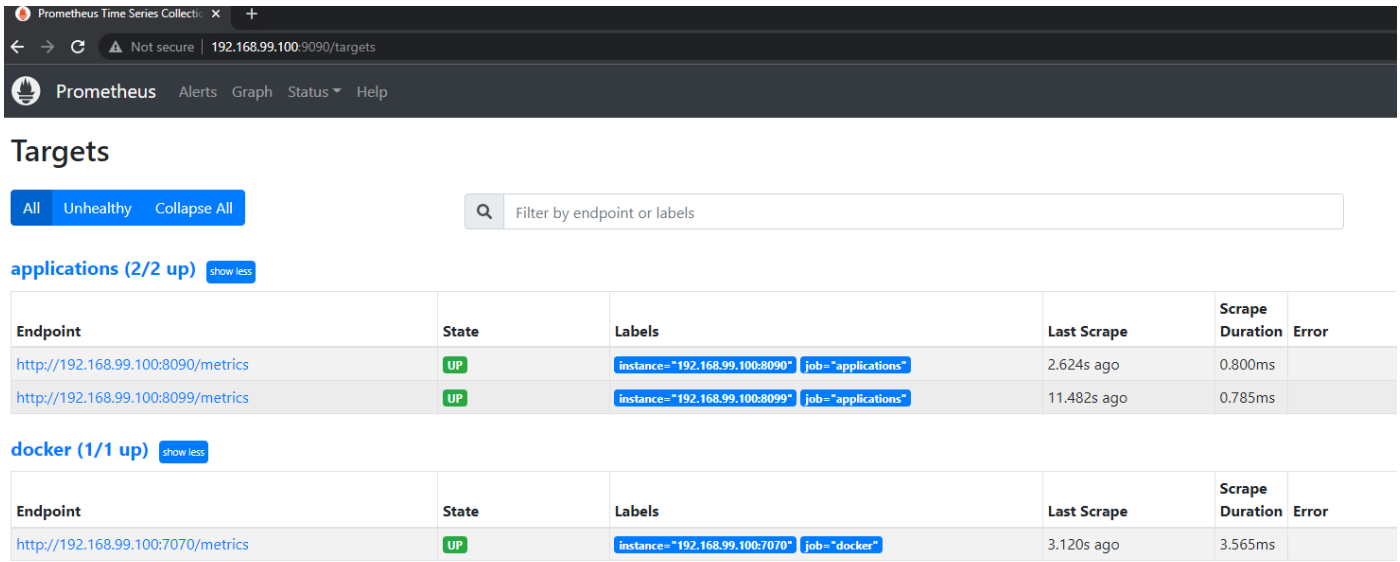
```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'docker'
    static_configs:
    - targets: ['192.168.99.100:7070']  # docker metrics
  - job_name: 'applications'
    static_configs:
    - targets: ['192.168.99.100:8090'] # goprom app 1 metrics
    - targets: ['192.168.99.100:8099'] # goptom app 2 metrics
```

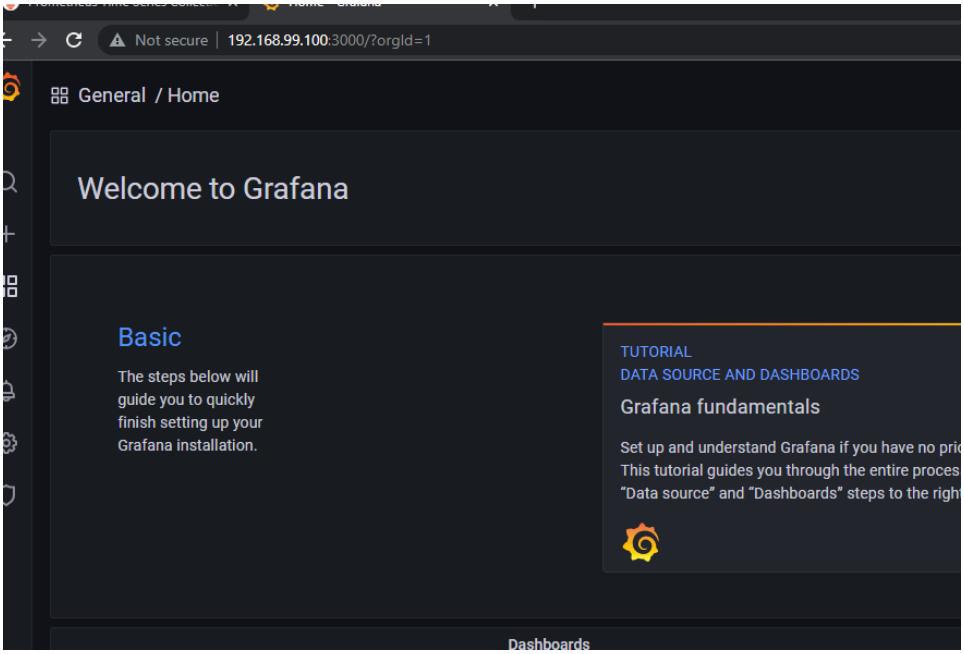
- Running Vagrant up, checking for src code in Node vm



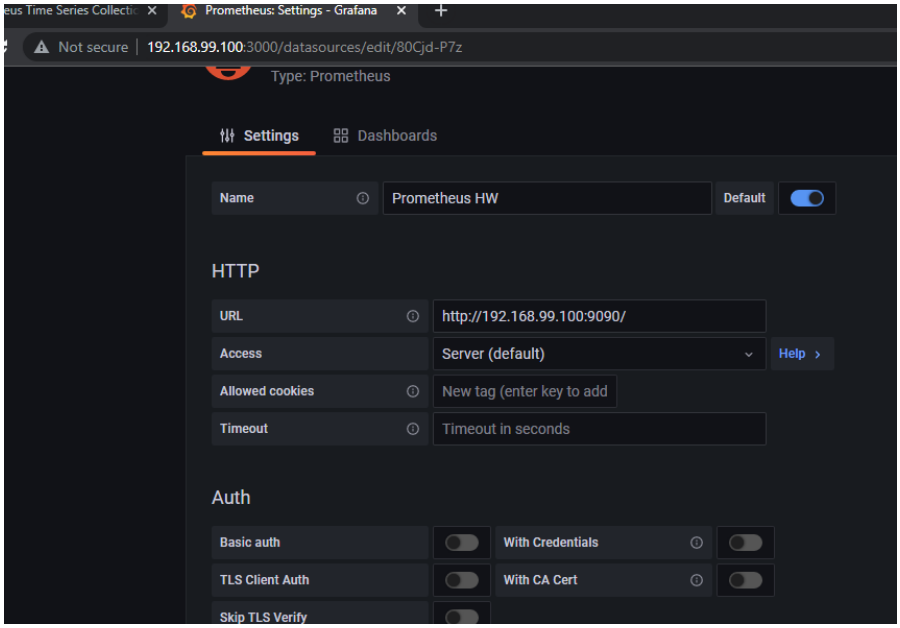
- Checking Prometheus Targets



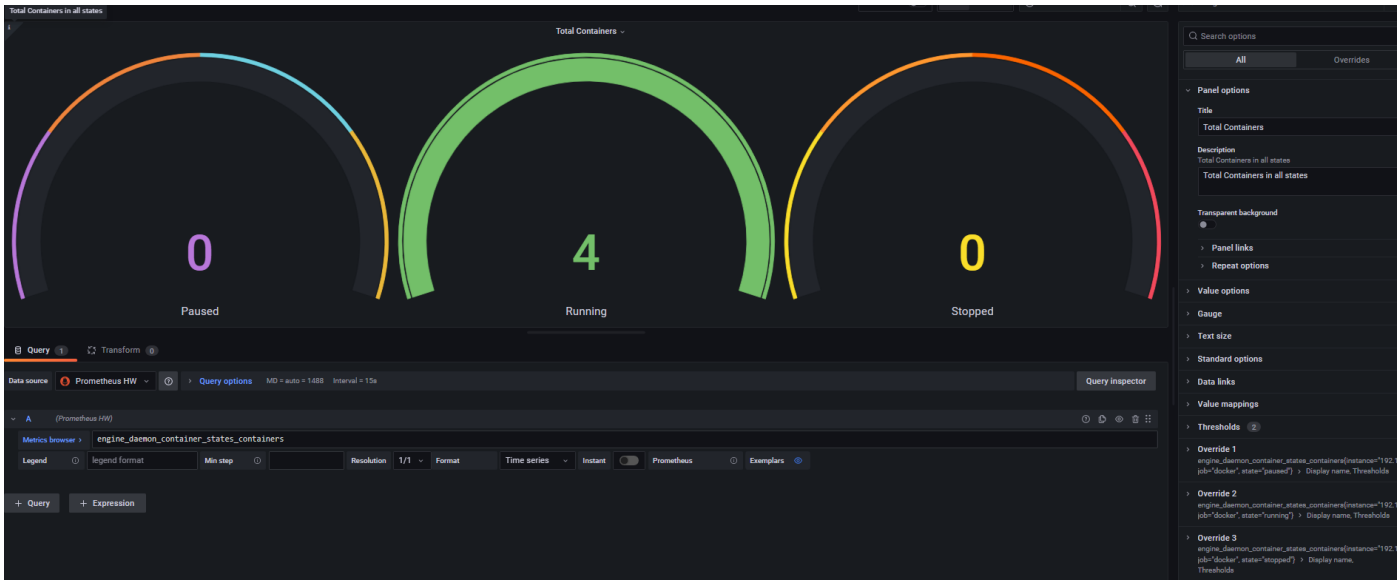
- Logging in to Grafana



- Adding Data Source to Grafana

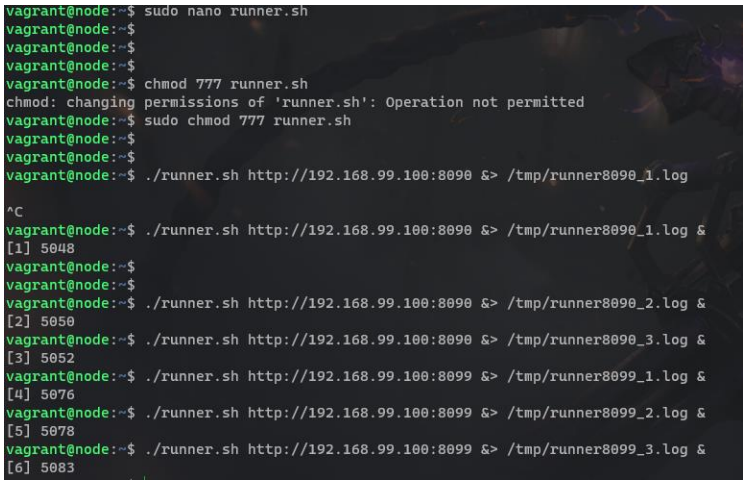


- Creating panel - A panel which shows how many containers are on the host (in all states)



- Adding runner.sh (from lab exercise) and starting some instances in order to generate some traffic to goprom apps.

NOTE: I forgot to add this in vagrant shared folder 😐



- Creating panel - A panel which shows how many jobs are processed by each goprom application (all result types)

- Stopping one container

```
vagrant@node:~$ docker container ls
CONTAINER ID   IMAGE              COMMAND                  CREATED             STATUS             PORTS
4b37528f0a34   src_goprom_node2   "/bin/goprom"            About an hour ago   Up About an hour   0.0.0.0:8099->8080/t
531e487a4019   src_goprom_node1   "/bin/goprom"            About an hour ago   Up About an hour   0.0.0.0:8090->8080/t
9652240e9436   grafana/grafana    "/run.sh"                About an hour ago   Up About an hour   0.0.0.0:3000->3000/t
0acdcafdeab1   prom/prometheus    "/bin/prometheus --c…"   About an hour ago   Up About an hour   0.0.0.0:9090->9090/t
vagrant@node:~$ docker stop 4b37528f0a34
4b37528f0a34
vagrant@node:~$ docker ps -a
CONTAINER ID   IMAGE              COMMAND                  CREATED             STATUS                       PORTS
4b37528f0a34   src_goprom_node2   "/bin/goprom"            About an hour ago   Exited (2) 4 seconds ago
531e487a4019   src_goprom_node1   "/bin/goprom"            About an hour ago   Up About an hour             0.0.0.0:8090
9652240e9436   grafana/grafana    "/run.sh"                About an hour ago   Up About an hour             0.0.0.0:3000
0acdcafdeab1   prom/prometheus    "/bin/prometheus --c…"   About an hour ago   Up About an hour             0.0.0.0:9090
```

- Checking Grafana Dashboard