Instalacja hadoopa

Środowisko

Przygotowaliśmy vagrantfile konfigurujący środowisko:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
  config.vm.network "private_network", ip:"192.168.56.10"

  config.vm.provider "virtualbox" do |v|
    v.memory = 4096
    v.cpus = 4
  end

  config.vm.network "forwarded_port", guest: 8088, host: 8088
  config.vm.network "forwarded_port", guest: 9870, host: 9870
  config.vm.box = "bento/ubuntu-20.04"
  config.vm.provision "shell", path: "init.sh"
end
```

Init.sh:

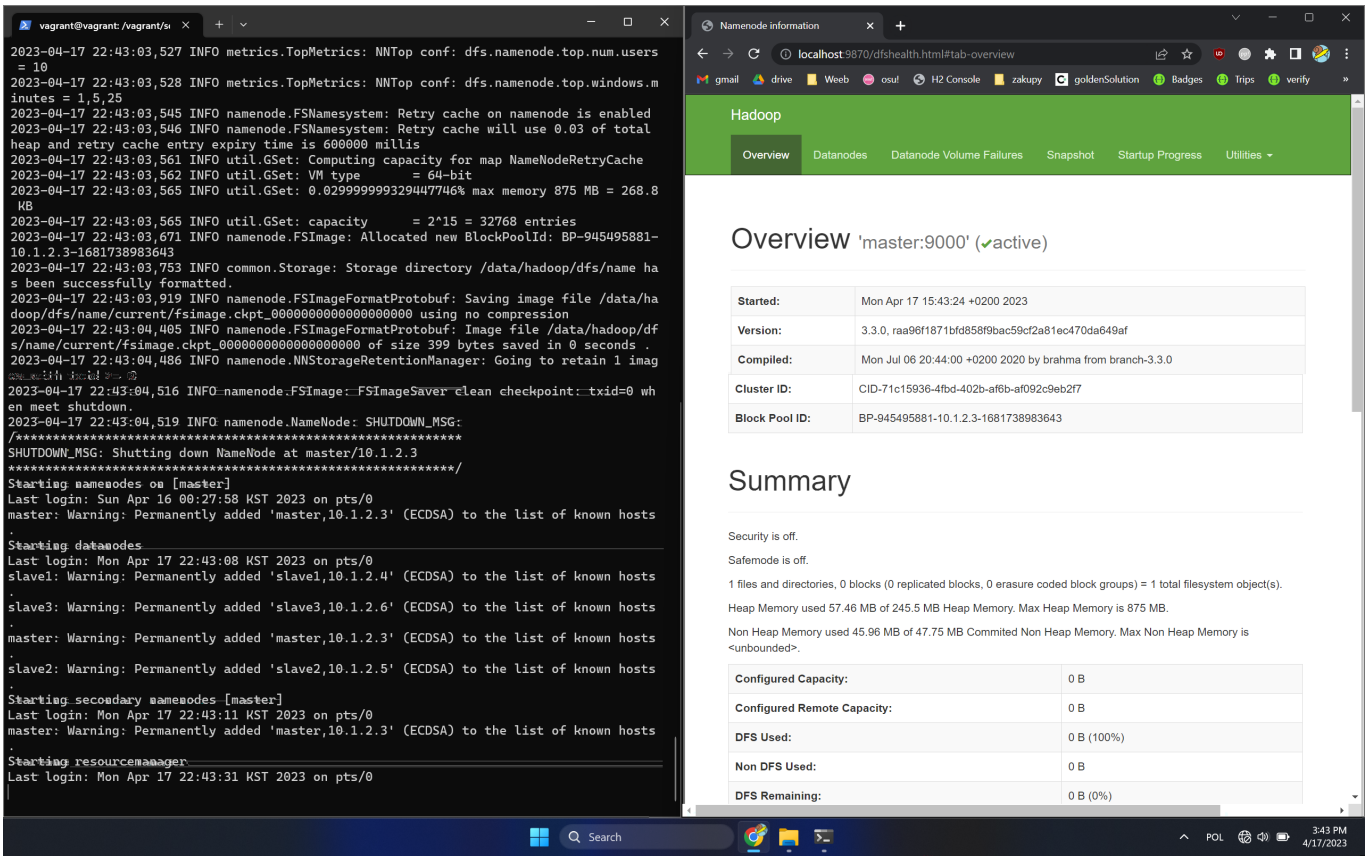
```
apt update
s
curl -fsSL https://get.docker.com -o get-docker.sh
sh get-docker.sh

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 groupadd docker
sudo usermod -aG docker vagrant
newgrp docker

sudo apt install -y python3
sudo apt install -y python3-pip
pip3 install docker
```

Przygotowaliśmy skrypt tworzący obrazy dockerowe, który należy ręcznie uruchomić za pierwszym razem kiedy maszyna jest utworzona:

```
./compose-up.sh 3.3.0 3 /tmp/hadoop /tmp/hadoop_logs /tmp/hbase_logs
/tmp/hive_logs /tmp/sqoop_logs mariadb /Users/Shared/workspace/docker-ws/maria-
data
```

Uruchomienie hadoopa

Hadoopa uruchamiamy zgodnie z instrukcją skryptem ./hadoop_start



Uruchomienie map-reduce na klastrze

Przygotowaliśmy projekt maven w IntelliJ. Po wygenerowaniu jara możemy umieścić go i uruchomić na klastrze za pomocą własnego skryptu w języku python:

```
#!/usr/bin/env python3

import docker
import os
import tarfile
import sys

client = docker.from_env()
filePath = sys.argv[1]
params = sys.argv[2:]
```

```
def copy_to(src, dst):
    name, dst = dst.split(':')
    container = client.containers.get(name)
    src = os.path.abspath(src)
    os.chdir(os.path.dirname(src))
    srcname = os.path.basename(src)
    tar = tarfile.open(src + '.tar', mode='w')
    try:
        tar.add(srcname)
    finally:
        tar.close()
    data = open(src + '.tar', 'rb').read()
    container.put_archive(os.path.dirname(dst), data)


copy_to(filePath, "master:/home/test.jar")

container = client.containers.get('master')
res = container.exec_run("yarn jar /home/test.jar " + " ".join(params))
print(res)
```