Vagrant
Docker
Bash

SKILLFACTORY

Содержание

- Vagrant Extra— Disk, Network, Multi Machine
- Docker Extra — Environment, Quota, Network
- Bash — Vagrant Provision

Vagrant Disk

Типы дисков:

- ➔ disk
- ➔ dvd
- ➔ floppy

```
config.vm.disk :disk, name: "backup", size: "10GB",  primary: true

config.vm.disk :dvd, name: "installer", file: "./installer.iso"

config.vm.disk :floppy, name: "cool_files"
```

Vagrant Disk

Изменение размера основного диска

```
Vagrant.configure("2") do |config|
  config.vm.define "hashicorp" do |h|
  h.vm.box = "ubuntu/jammy64"

  h.vm.provider :virtualbox
   h.vm.disk  :disk,  size:  "100GB",  primary:
true
  end
end
```

Vagrant Disk

Подключение новых дисков

```
Vagrant.configure("2") do |config|
     config.vm.define "hashicorp" do |h|
          h.vm.box = "ubuntu/jammy64"

          h.vm.provider :virtualbox
          (0..3).each do |i|
                    h.vm.disk :disk, size: "5GB", name: "disk-#{i}"
          end
     end
end
```

Vagrant Disk

Подключение оптических дисков

```
Vagrant.configure("2") do |config|
  config.vm.define "hashicorp" do |h|
          h.vm.box = "hashicorp/bionic64"

          h.vm.provider :virtualbox
          h.vm.disk :dvd, name: "installer", file: "./installer.iso"
  end
end
```

Vagrant Network

Имя хоста

```
Vagrant.configure("2") do |config|
    # ...
  config.vm.hostname = "myhost.local"
end
```

Vagrant Network

Проброс портов

```
Vagrant.configure("2") do |config|
    # ...
  config.vm.network "forwarded_port", guest: 80, host: 8080
end
```

«forwarded_port» - идентификатор сети

Vagrant Network

Проброс портов: протокол

```
Vagrant.configure("2") do |config|
   # ...
  config.vm.network "forwarded_port", guest: 2003, host: 12003, protocol: "tcp"
  config.vm.network "forwarded_port", guest: 2003, host: 12003, protocol: "udp"
end
```

«forwarded_port» - идентификатор сети

Vagrant Network

Проброс портов: исправление коллизии

```
Vagrant.configure("2") do |config|
   # ...
   config.vm.network "forwarded_port", guest: 80, host: 8080, auto_correct: true end
```

«forwarded_port» - идентификатор сети

Vagrant Network

Диапазон портов для разрешения коллизии

```
Vagrant.configure("2") do |config|
   # ...
   config.vm.usable_port_range = 8000..8999
 end
```

Vagrant Network

Частная сеть: DHCP

```
Vagrant.configure("2") do |config|
   # ...
   config.vm.network "private_network", type: "dhcp"
 end
```

«private_network» - идентификатор сети

Vagrant Network

Частная сеть: IPv6.

```
Vagrant.configure("2") do |config|
   # ...
        config.vm.network "private_network",
                ip: "fde4:8dba:82e1::c4",
                netmask: "96"
 end
```

Значение netmask по умолчанию: 64

Vagrant Network

Частные IP адреса

Диапазон	Количество	Маска
10.0.0.0 – 10.255.255.255	16 777 216	10.0.0.0/8 (255.0.0.0)
172.16.0.0 – 172.31.255.255	1 048 576	172.16.0.0/12 (255.240.0.0)
192.168.0.0 – 192.168.255.255	65 536	192.168.0.0/16 (255.255.0.0)

Docker по умолчанию использует сеть 172.17.0.0/16

Vagrant Network

Публичная сеть: статический IP

```
Vagrant.configure("2") do |config|
   # ...
   config.vm.network "public_network", ip: “192.168.0.17”
 end
```

«public_network» - идентификатор сети

Vagrant Network

Публичная сеть: интерфейс

```
Vagrant.configure("2") do |config|
   # ...
        config.vm.network "public_network", bridge: [
              "en1: Wi-Fi (AirPort)",
              "en6: Broadcom NetXtreme Gigabit Ethernet Controller"]
 end
```

«public_network» - идентификатор сети

Vagrant Multi Machine

```
Vagrant.configure("2") do |config|
        config.vm.provision "shell", inline: "echo Hello"

        config.vm.define "web" do |web|
                web.vm.box = "apache"
        end

        config.vm.define "db" do |db|
                db.vm.box = "mysql"
        end
end
```

Vagrant Multi Machine

Основная машина

```
Vagrant.configure("2") do |config|
   # ...
        config.vm.define "web", primary: true do |web|
  # ...
        end
end
```

Vagrant Multi Machine

Linked Clone

```
config.vm.provider "virtualbox" do |vb|

   vb.linked_clone = true
   vb.cpus = "2"
   vb.memory = "2048"

 end
```

Содержание

- Vagrant Extra— Disk, Network, Multi Machine
- Docker Extra — Environment, Quota, Network
- Bash — Vagrant Provision

Docker Environment

app.env

```
MY_SECRET_KEY=SOME_SECRET
IMAGE_NAME=docker_image
```

docker-compose.yml

```
services:
  api:
    image: "${IMAGE_NAME}"
    env_file:
     - ./app.env
    environment:
     - NODE_ENV=production
     - MY_SECRET_KEY: ${MY_SECRET_KEY}
```

.env

Каждая линия представляет пару ключ-значение.

- ➔ VAR=VAL -> VAL
- ➔ VAR="VAL“ -> VAL
- ➔ VAR=‘VAL’ -> VAL

Комментарии начинаются с пробела.

- ➔ VAR=VAL # comment -> VAL
- ➔ VAR=VAL# no comment -> VAL# not a comment
- ➔ VAR="VAL # no comment“ -> VAL # no comment
- ➔ VAR="VAL" # comment -> VAL

.env

Escape символы:

- ➔ VAR="some\tvalue"-some value
- ➔ VAR='some\tvalue'-> some\tvalue
- ➔ VAR=some\tvalue -> some\tvalue

Экранирование:

- ➔ VAR='$OTHER’ -> $OTHER
- ➔ VAR='${OTHER}’-> ${OTHER}
- ➔ VAR='Let\'s go!’ -> Let's go!
- ➔ VAR="{\"hello\": \"json\"}“ -> {"hello": "json"}

Docker-compose Environment

Прямая замена:

➔ **${VAR}** -> значение **VAR**

Значение по умолчанию

➔ **${VAR:-default}** -> значение, VAR если установлено и не пусто, иначе default

➔ **${VAR-default}** -> значение VAR если установлено, иначе default

Требуемое значение:

➔ **${VAR:?error}** -> значение, **VAR** если установлено и не пусто, иначе **ошибка**

➔ **${VAR?error}** -> значение **VAR** если установлено, иначе **ошибка**

Docker-compose Environment

Альтернативное значение:

- ➔ **${VAR:+replacement} -> replacement** если **VAR** установлено и не пусто, иначе пусто
- ➔ **${VAR+replacement}**-> **replacement** если **VAR** установлено, иначе пусто

Docker Compose Network

Проброс портов

```
services:
   web:
     build: .
     ports:
          - "8000:8000"
    db:
     image: postgres
     ports:
          - "8001:5432"
```

Docker Compose Network

Доп. сеть

```
services:
  web:
    build: .
    networks:
     -  mynet
     -  backend
```

docker-compose.yml

```
networks:
  mynet1:
   ipam:
    driver: default
    config:
        - subnet: 172.28.0.0/16
          ip_range: 172.28.5.0/24
          gateway: 172.28.5.254
```

Network Driver

- ➔ **bridge** – сетевой драйвер по умолчанию
- ➔ **host** – использование сети хоста напрямую
- ➔ **overlay** – сеть для объединения нескольких демонов Docker
- ➔ **ipvlan** – сеть с контролем адресации IPv4, так и IPv6
- ➔ **macvlan** – сеть с маршрутизацией трафика по MAC - адресам

Docker Compose Volumes

Конфигурация тома NFS

```
volumes:
  example:
   driver_opts:
    type: "nfs"
    o: "addr=10.40.0.199,nolock,soft,rw"
    device: ":/docker/example"
```

Docker Compose Quotas

```
cpus: 0.5 // количество ядер CPU
cpu_shares: 73 // количество циклов CPU
cpu_quota: 50000 // количество циклов использования CPU в период  cpu_period
cpu_period:  20ms  //  продолжительность  времени  использования  CPU  в
миллисекундах
cpuset: 0,1 // указание конкретных CPU которые может использовать контейнер
```

Docker Compose Quotas

```
mem_limit: 1000m // жесткое ограничение
memswap_limit: 20000m // общее количество памяти ОЗУ+подкачка
mem_reservation: 512m  //мягкое ограничение
```

Содержание

- Vagrant Extra— Disk, Network, Multi Machine
- Docker Extra — Environment, Quota, Network
- Bash — Vagrant Provision

Vagrant Provisioner

Копирование файла

```
Vagrant.configure("2") do |config|
   # ...
   config.vm.provision "file", source: "~/.gitconfig", destination: ".gitconfig"
 end
```

Vagrant Provisioner

Shell: inline scripts

```
Vagrant.configure("2") do |config|
   # ...
   config.vm.provision "shell", inline: "echo Hello, World"
 end
```

Vagrant Provisioner

Shell: inline scripts

```
$script = <<-SCRIPT
echo I am provisioning... date > /etc/vagrant_provisioned_at
SCRIPT

Vagrant.configure("2") do |config|
   # ...
   config.vm.provision "shell", inline: $script
 end
```

Vagrant Provisioner

Shell: external script

```
Vagrant.configure("2") do |config|
    # ...
    config.vm.provision "shell",
path: "https://example.com/provisioner.sh"

end
```

Vagrant Provisioner

Shell: script arguments

```
Vagrant.configure("2") do |config|
    # ...
    config.vm.provision "shell" do |s|
  s.inline = "echo $1"
  s.args = ["hello, world!"]
  end
end
```

Vagrant Provisioner

Синхронизация директорий

```
Vagrant.configure("2") do |config|
   # ...
   config.vm.synced_folder ".", "/vagrant", owner: "vagrant",
group: "vagrant",   mount_options: ["uid=1234", "gid=1234"]
end
```

Ссылки

- Vagrant Disk
- Vagrant Disk VirtualBox
- Проброс портов
- Приватная сеть
- Публичная сеть
- VirtualBox Networking
- File Provisioner

Ссылки

- ➔ Shell Provisioner
- ➔ Multi-Machine
- ➔ Self Hosted Vagrant Cloud
- ➔ Docker Environment
- ➔ Docker Networking
- ➔ Docker Volumes
- ➔ Compose file
- ➔ Privilege and Linux capabilities