Vagrant box setup

Artur Malinowski (PGS Software)

# Creating custom vagrant box

Install Oracle VirtualBox and Vagrant  
In any folder of your choice, create configuration file named Vagrantfile   
C:\Users\amalinowski\Desktop\vagrantfile.png  
  
Insert code fragment from below into Vagrantfile   
Some explanation:

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

config.vm.box = "ubuntu/trusty64"

config.vm.provision :shell, path: "provision.sh"

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

vb.customize ["modifyvm", :id, "--memory", "1024"]

end

end

tells Vagrant which box should be loaded, ubuntu/trusty64 is the name of basic box provided by HashiCorp. If it is not available locally, Vagrant will download it from Vagrant’s public box catalog ([*https://atlas.hashicorp.com/boxes/search*](https://atlas.hashicorp.com/boxes/search)) when we try to run our box.

config.vm.box = "ubuntu/trusty64"

points to shell script which will be executed after vagrant box is up and running. Content of provision.sh will be explained later on.

config.vm.provision :shell, path: "provision.sh"

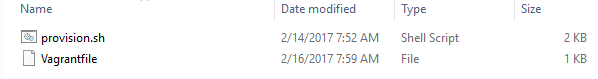
config.vm.provider "virtualbox" do |vb|

vb.customize ["modifyvm", :id, "--memory", "1024"]

end

sets memory limit for vagrant box to 1024 MB

Create provision.sh in the same folder as Vagrantfile



Insert code fragment from below into provision.sh

#!/bin/sh -e

#POSTGRES

echo "Installing PostreSQL"

# PostgreSQL Version

PG\_VERSION=9.4

export DEBIAN\_FRONTEND=noninteractive

PG\_REPO\_APT\_SOURCE=/etc/apt/sources.list.d/pgdg.list

# Add PG apt repo:

echo "deb http://apt.postgresql.org/pub/repos/apt/ trusty-pgdg main" > "$PG\_REPO\_APT\_SOURCE"

# Add PGDG repo key:

wget --quiet -O - https://apt.postgresql.org/pub/repos/apt/ACCC4CF8.asc | apt-key add -

# Update package list and upgrade all packages

apt-get update

apt-get -y upgrade

apt-get -y install "postgresql-$PG\_VERSION" "postgresql-contrib-$PG\_VERSION"

PG\_CONF="/etc/postgresql/$PG\_VERSION/main/postgresql.conf"

PG\_HBA="/etc/postgresql/$PG\_VERSION/main/pg\_hba.conf"

PG\_DIR="/var/lib/postgresql/$PG\_VERSION/main"

# Edit postgresql.conf to change listen address to '\*':

sed -i "s/#listen\_addresses = 'localhost'/listen\_addresses = '\*'/" "$PG\_CONF"

# Append to pg\_hba.conf to add password auth:

echo "host all all all md5" >> "$PG\_HBA"

# Explicitly set default client\_encoding

echo "client\_encoding = utf8" >> "$PG\_CONF"

service postgresql restart

echo "Successfully created PostgreSQL dev virtual machine."

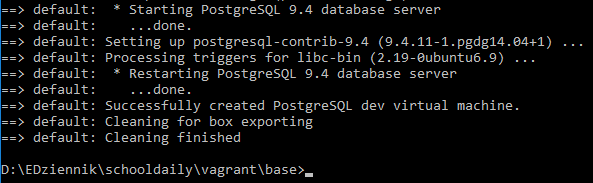
echo "Cleaning for box exporting"

sudo apt-get clean

echo "Cleaning finished"

provision.sh script downloads and installs PostgreSQL version 9.4.  
Script content will be explained in further version of this guide.

Next step is to start box based on Vagrantfile. Open command line, go to folder with Vagrantfile and execute vagrant up command.   
Screen after vagrant start up:

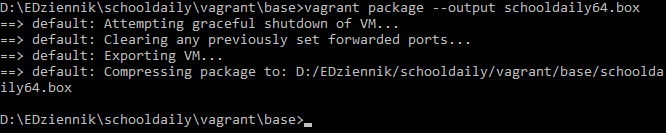


Now vagrant basic box is ready for packaging to final box.

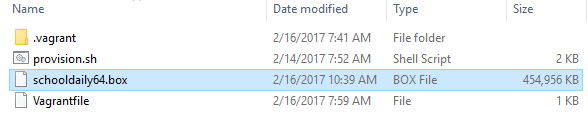
In command line type in:

vagrant package –output schooldaily64.box

Packaging may take a while…  
Screen after packaging:



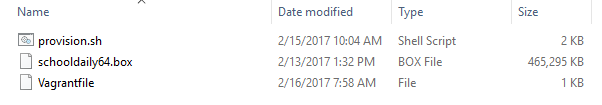
Vagrant should create schooldaily64.box file:



And that’s it, our custom box with preinstalled PostgreSQL is ready to use.

# starting custom vagrant box

In new empty folder create Vagrantfile, provision.sh and copy newly created schooldaily64.box.



Insert code fragment from below into Vagrantfile

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

config.vm.box = "schooldaily64"

config.vm.box\_url = "file://schooldaily64.box"

config.vm.provision :shell, path: "provision.sh"

config.vm.network :forwarded\_port, guest:5432, host:15432

end

sets name of registered vagrant box to use, if schooldaily64 is not registered by vagrant then box will be imported from config.vm.box\_url

config.vw.box = “schooldaily64”

points to box file which should be imported and registered by vagrant when there is no schooldaily64 box available in vagrant local cache. It can also point to web resources i.e.:   
config.vm.box\_url = “https://atlas.hashicorp.com/ubuntu/boxes/trusty64”

config.vm.box\_url = “file://schooldaily64.box”

points to shell script that will be executed after vagrant box is up and running.

config.vm.provision :shell, path: "provision.sh"

forwards PostgreSQL default access port to be accessible from outside the VM on port 15432

config.vm.network :forwarded\_port, guest:5432, host:15432

Insert code fragment from below into provision.sh

echo "Starting PostgreSQL"

APP\_DB\_USER=pgsuser

APP\_DB\_PASS=pgspass

# Database Name

APP\_DB\_NAME=schooldaily

print\_db\_usage () {

echo " PostgreSQL database has been setup and can be accessed on port 15432"

echo " Host: localhost"

echo " Port: 15432"

echo " Database: $APP\_DB\_NAME"

echo " Username: $APP\_DB\_USER"

echo " Password: $APP\_DB\_PASS"

echo ""

echo "Admin access to postgres user via VM:"

echo " vagrant ssh"

echo " sudo su - postgres"

echo ""

echo "psql access to app database user via VM:"

echo " vagrant ssh"

echo " sudo su - postgres"

echo " PGUSER=$APP\_DB\_USER PGPASSWORD=$APP\_DB\_PASS psql -h localhost $APP\_DB\_NAME"

}

cat << EOF | su - postgres -c psql

-- Create the database user:

CREATE USER $APP\_DB\_USER WITH PASSWORD '$APP\_DB\_PASS';

-- Create the database:

CREATE DATABASE $APP\_DB\_NAME WITH OWNER=$APP\_DB\_USER

LC\_COLLATE='en\_US.utf8'

LC\_CTYPE='en\_US.utf8'

ENCODING='UTF8'

TEMPLATE=template0;

EOF

sudo su postgres

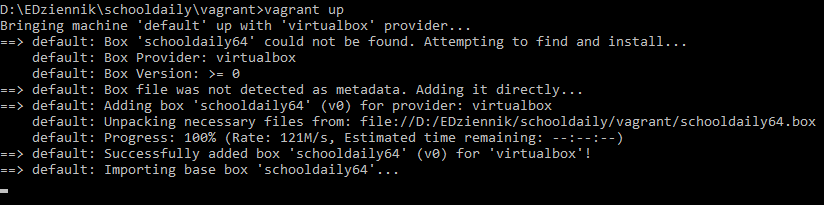
service postgresql restart

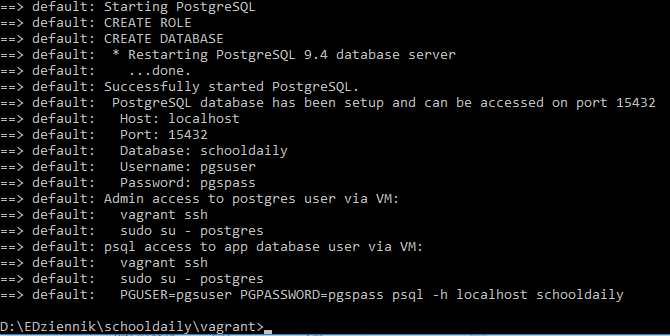
echo "Successfully started PostgreSQL."

print\_db\_usage

provision.sh creates new PostgreSQL user *pgsuser* with password *pgspass* and *schooldaily* database. It also restarts PostgreSQL and display some additional information about accessing it.

Now we can call vagrant up





On final screen, there are some information about how to access PostgreSQL on our VM.