Answers to Self-check Questions section

Danis Alukaev B19-DS-01

1. What will happen if you will start the vagrant machine without `$vagrant init`. Why?

   This command is used to initialize the current directory as a Vagrant environment. It is done by creating a Vagrantfile (if it does not exist) containing information about type, configuration, and provision of the machine.
   As long as the directory contains Vagrantfile, there should be no problem when the machine starts. However, if it does not contain this file and the command has not been executed, an error about the absence of an environment is likely to occur. For example, on my PC this error is following:

   A Vagrant environment or target machine is required to run this command. Run `vagrant init` to create a new Vagrant environment. Or, get an ID of a target machine from `vagrant global-status` to run this command on. A final option is to change to a directory with a Vagrantfile and to try again.

   Indeed, the problem is that Vagrant cannot find the description of the machine to be created that has to be in Vagrantfile.

2. To bring up the Vagrant virtual environment, you can use `$vagrant up`. What will happen after the command?

   After this command, the Vagrant initializes the building of guest machines based on the Vagrantfile. For the simplest case, it firstly attempts to find and install a specified box, e.g. `ubuntu/bionic64`. Secondly, it will configure all networking interfaces and forward the ports. Once the machine is booted it inserts a new SSH key. Thirdly, the Vagrant will check for guest additions in the virtual machine. Finally, it mounts a shared folder.

3. How do you access (login into) virtual machine created with vagrant?

   In order to access running virtual machines one could use the command `vagrant ssh`. This will give access to the shell interface. Optionally, one can specify the name or id of the machine that should be accessed.

4. What is a BOX in Vagrant?

Essentially, the concept of box can be described as a template for a Vagrant environment. This approach allows sharing snapshots of the system between multiple users to bring up identical environments. There is a public list of available Vagrant environments, so that managing and using them becomes quite handy.

5. What is Provider in Vagrant?

   This is the product performing virtualization. The Vagrant supports various applications, e.g. VirtualBox, Hyper-V, Docker and others. Although it does make sense to use alternate providers. For instance, the VMware is recommended in official documentation for development process as a more stable and performant in comparison with VirtualBox.

6. What is Provisioner in Vagrant?

   Provisioners allow to automate the routine processes, e.g. alternating configurations, installation of software, etc. Instead of doing this by hand, all the job is carried out on the `vagrant up` stage. Provisioners can be implemented in the form of shell scripts or industry-standard configuration management systems.

7. What is Vagrantfile?

   The Vagrantfile contains guidelines to create a guest machine, i.e., information about type, configuration, and provision of the machine.

8. What are Synced Folders in Vagrant?

   Synced folders are the directories used to share data between the guest and host machines. The default shared folder is the working directory on the host machine mounted to the folder /vagrant on the guest machine.

9. What is Multi-Machine environment in Vagrant?

   The Vagrant is capable of controlling multiple guest machines. The goal is to create an infrastructure composed of machines somehow associated with each other. It can be used for testing the multi-server topology, modeling distributed systems and other.

10. How do you define multiple machines in Vagrant?

The simplest example of Vagrantfile configuration is following:

```
BOX_IMAGE = "ubuntu/bionic64"

Vagrant.configure("2") do |config|
  config.vm.define "master" do |subconfig|
    subconfig.vm.box = BOX_IMAGE
  end

  config.vm.define "node" do |subconfig|
    subconfig.vm.box = BOX_IMAGE
  end
end
```

11. What does `vagrant push` do?

    This command allows pushing or deploying of application code to a remote host, e.g. FTP server. Pushes are defined in Vagrantfile.

12. What does `vagrant box list` do?

    This command reports all Vagrant boxes installed.

13. What does `vagrant box outdated` do?

    This command checks if the currently used Vagrant environment is outdated. It shows the latest versions for available providers.

14. What does `vagrant box prune` do?

    This command removes previous versions of installed Vagrant boxes.

15. What does `vagrant box remove` do?

    This command removes the Vagrant box with a given name. If several providers are used or several versions available, then it is necessary to specify the particular one.

16. What does `vagrant box repackage` do?

This command creates a `.box` file in a local directory that can be distributed.

17. What does `vagrant box update` do?

    This command updates the box used for the current Vagrant environment if necessary. Important note is that changes will not reflect in the running environment, so one should destroy it and bring it back up.

18. What does `vagrant connect` do?

    There is another command `vagrant share` that is used to share a Vagrant environment with other users. It allows direct collaboration ignoring the network environment. Accordingly, the command `vagrant connect` is used to connect to remove the Vagrant environment.

19. What does `vagrant destroy` do?

    This command stops the running Vagrant machine and removes all resources used. It is helpful, because after execution the PC comes to its original appearance, as if the Vagrand was never used.