

实践项目环境搭建和测试

1. 新建目录 解压文件

将project.zip解压到一个**不包含中文的路径**的目录, 如 D:\Projects\netProj

然后解压目录中的tju_tcp.zip, 最终形成如图所示结构

此电脑 > 固态1 (G:) > Projects > netProj				
名称	修改日期	类型	大小	
tju_tcp	2021/8/14 15:28	文件夹		
tju_tcp.zip	2021/8/14 15:28	zip Archive	78 KB	
ubuntu_netproj.box	2021/8/10 21:13	BOX 文件	1,128,033...	
vagrant_2.2.18_x86_64.msi	2021/8/10 11:13	Windows Install...	259,200 KB	
Vagrantfile	2021/8/10 21:31	文件	2 KB	
VirtualBox-6.1.26-145957-Win.exe	2021/8/10 11:10	应用程序	105,699 KB	

- **tju_tcp目录:** 本次实践的工作目录 存放所有源码
- **ubuntu_netproj.box:** 专为本次实践搭建的虚拟镜像文件 本身是Ubuntu20.04服务器版 安装了所有必须的依赖和库
- **vagrant_2.2.18_x86_64.msi:** vagrant软件的安装包
- **Vagrantfile:** vagrant启动虚拟机所用的配置文件 内部定义了所用镜像, 网络环境, 目录挂载等信息
- **VirtualBox-6.1.26-145957-Win.exe:** VirtualBox的安装包 vagrant后台是使用VirtualBox运行虚拟机的

2. 安装Oracle VM VirtualBox和Vagrant软件

运行两款软件的安装程序, 除安装目录可自定义外, 其余均按照默认设置进行安装

安装过程中可能会请求安装一些辅助插件, 要同意请求, 进行安装

安装完毕后按照要求重启计算机

安装完成后, 在CMD输入vagrant命令, 应该得到如下输出

```

PS Usage: vagrant [options] <command> [<args>]                                vagrant

-h, --help                                Print this help.

Common commands:
  autocomplete  manages autocomplete installation on host
  box           manages boxes: installation, removal, etc.
  cloud        manages everything related to Vagrant Cloud
  destroy      stops and deletes all traces of the vagrant machine
  global-status outputs status Vagrant environments for this user
  halt         stops the vagrant machine
  help        shows the help for a subcommand
  init        initializes a new Vagrant environment by creating a Vagrantfile
  login
  package     packages a running vagrant environment into a box
  plugin      manages plugins: install, uninstall, update, etc.
  port        displays information about guest port mappings
  powershell  connects to machine via powershell remoting
  provision   provisions the vagrant machine
  push        deploys code in this environment to a configured destination
  rdp         connects to machine via RDP
  reload      restarts vagrant machine, loads new Vagrantfile configuration
  resume      resume a suspended vagrant machine
  snapshot    manages snapshots: saving, restoring, etc.
  ssh         connects to machine via SSH
  ssh-config  outputs OpenSSH valid configuration to connect to the machine
  status      outputs status of the vagrant machine
  suspend     suspends the machine
  up          starts and provisions the vagrant environment
  upload      upload to machine via communicator
  validate    validates the Vagrantfile
  vbguest     plugin: vagrant-vbguest: install VirtualBox Guest Additions to the machine
  version     prints current and latest Vagrant version
  winrm       executes commands on a machine via WinRM
  winrm-config outputs WinRM configuration to connect to the machine

For help on any individual command run `vagrant COMMAND -h`

Additional subcommands are available, but are either more advanced
or not commonly used. To see all subcommands, run the command
`vagrant list-commands`.
  --[no-]color          Enable or disable color output
  --machine-readable    Enable machine readable output
  -v, --version         Display Vagrant version
  --debug              Enable debug output
  --timestamp          Enable timestamps on log output
  --debug-timestamp    Enable debug output with timestamps
  --no-tty             Enable non-interactive output

```

3. 添加box 安装vagrant插件

```

1 # 确保目前位于项目根目录
2 cd D:\Projects\netProj
3
4 # 使用如下命令安装vagrant-vbguest插件 该插件能协助处理虚拟机和物理机之间的文件共享
5 vagrant plugin install --plugin-clean-sources --plugin-source
  https://gems.ruby-china.com/ vagrant-vbguest
6
7 # 使用如下命令向vagrant中添加box 可以理解该box是一个已经设置完毕的虚拟机镜像 我们的项目将
  在这个box中运行
8 vagrant box add ubuntu/netproj .\ubuntu_netproj.box
9

```

4. 开启虚拟机

```

1 # 使用如下命令开启虚拟环境
2 vagrant up

```

如果正常开启应该看到如下输出

```

PS G:\Projects\netProj> vagrant up
Bringing machine 'client' up with 'virtualbox' provider...
Bringing machine 'server' up with 'virtualbox' provider...
=> client: Importing base box 'ubuntu/netproj'...
=> client: Matching MAC address for NAT networking...
=> client: Setting the name of the VM: netProj_client_1628927454935_45216
=> client: Clearing any previously set network interfaces...
=> client: Preparing network interfaces based on configuration...
client: Adapter 1: nat
client: Adapter 2: intnet
=> client: Forwarding ports...
client: 22 (guest) => 2222 (host) (adapter 1)
=> client: Booting VM...
=> client: Waiting for machine to boot. This may take a few minutes...
client: SSH address: 127.0.0.1:2222
client: SSH username: vagrant
client: SSH auth method: password
client: Warning: Connection aborted. Retrying...
client:
client: Inserting generated public key within guest...
client: Removing insecure key from the guest if it's present...
client: Key inserted! Disconnecting and reconnecting using new SSH key...
=> client: Machine booted and ready!
[client] GuestAdditions 6.1.26 running --- OK.
=> client: Checking for guest additions in VM...
=> client: Setting hostname...
=> client: Configuring and enabling network interfaces...
=> client: Mounting shared folders...
client: /vagrant => G:/Projects/netProj
client: /vagrant/tju_tcp => G:/Projects/netProj/tju_tcp
=> client: Running provisioner: shell...
client: Running: inline script
=> client: Running provisioner: shell...
client: Running: inline script
=> client: Running provisioner: shell...
client: Running: inline script
=> server: You assigned a static IP ending in ".1" to this machine.
=> server: This is very often used by the router and can cause the
=> server: network to not work properly. If the network doesn't work
=> server: properly, try changing this IP.
=> server: Importing base box 'ubuntu/netproj'...
=> server: Matching MAC address for NAT networking...
=> server: You assigned a static IP ending in ".1" to this machine.
=> server: This is very often used by the router and can cause the
=> server: network to not work properly. If the network doesn't work
=> server: properly, try changing this IP.
=> server: Setting the name of the VM: netProj_server_1628927531138_36966
=> server: Fixed port collision for 22 => 2222. Now on port 2200.
=> server: Clearing any previously set network interfaces...
=> server: Preparing network interfaces based on configuration...
server: Adapter 1: nat
server: Adapter 2: intnet
=> server: Forwarding ports...
server: 22 (guest) => 2200 (host) (adapter 1)
=> server: Booting VM...
=> server: Waiting for machine to boot. This may take a few minutes...
server: SSH address: 127.0.0.1:2200
server: SSH username: vagrant
server: SSH auth method: password
server: Warning: Connection aborted. Retrying...
server:
server: Inserting generated public key within guest...
server: Removing insecure key from the guest if it's present...
server: Key inserted! Disconnecting and reconnecting using new SSH key...
=> server: Machine booted and ready!
[server] GuestAdditions 6.1.26 running --- OK.
=> server: Checking for guest additions in VM...
=> server: Setting hostname...
=> server: Configuring and enabling network interfaces...
=> server: Mounting shared folders...
server: /vagrant => G:/Projects/netProj
server: /vagrant/tju_tcp => G:/Projects/netProj/tju_tcp
=> server: Running provisioner: shell...
server: Running: inline script
=> server: Running provisioner: shell...
server: Running: inline script
=> server: Running provisioner: shell...
server: Running: inline script

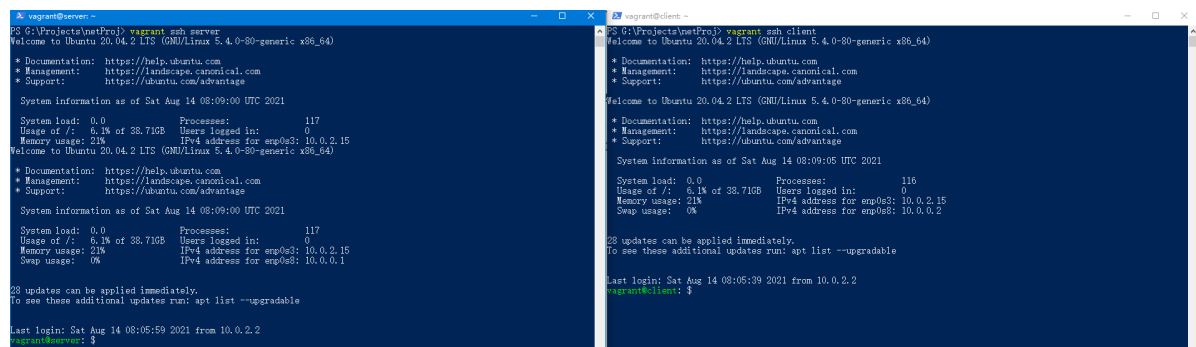
```

该命令会使用virtual box开启两个虚拟机，一个server 一个client

- 两台虚拟机均为Ubuntu 20.04服务器版本, 没有GUI
- 当前目录下的tju_tcp目录同时挂载在了两台虚拟机中的/vagrant/tju_tcp目录下
这意味着在windows下对tju_tcp目录的任何改动都会实时同步到两台虚拟机中
- 两台虚拟机都开启了ssh密码验证, 并且登录用户名和密码都为vagrant
server虚拟机ssh的端口为2200 client的端口为2222
- 两台虚拟机之间网络是互通的
server的IP地址为10.0.0.1 client的IP地址为10.0.0.2
两台虚拟机之间网络通讯进行了延迟和带宽的设置, 其中通信延迟为20ms, 最大带宽为100Mbps

5. 使用ssh连入server和client

- 1 # 在当前目录(.vagrant存在的目录)使用 **vagrant ssh server/client** 连入虚拟机
- 2 **vagrant ssh server** # 连入 **server** 虚拟机
- 3 **vagrant ssh client** # 连入 **client** 虚拟机



```
vagrant@server:~$ vagrant ssh server
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-80-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Sat Aug 14 08:09:00 UTC 2021

System load: 0.0          Processes: 117
Usage of /:  6.1% of 38.71GB Users logged in: 0
Memory usage: 21%        IPv4 address for enp0s3: 10.0.2.15
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-80-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Sat Aug 14 08:09:00 UTC 2021

System load: 0.0          Processes: 117
Usage of /:  6.1% of 38.71GB Users logged in: 0
Memory usage: 21%        IPv4 address for enp0s3: 10.0.2.15
Swap usage:  0%          IPv4 address for enp0s8: 10.0.0.1

28 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Last login: Sat Aug 14 08:05:59 2021 from 10.0.2.2
vagrant@server:~$
```

```
vagrant@client:~$ vagrant ssh client
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-80-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-80-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Sat Aug 14 08:09:05 UTC 2021

System load: 0.0          Processes: 116
Usage of /:  6.1% of 38.71GB Users logged in: 0
Memory usage: 21%        IPv4 address for enp0s3: 10.0.2.15
Swap usage:  0%          IPv4 address for enp0s8: 10.0.0.2

28 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Last login: Sat Aug 14 08:05:39 2021 from 10.0.2.2
vagrant@client:~$
```

当然, 也可以使用传统的ssh方法在任意终端中连接到虚拟机, 连接所需的信息已在上文给出, 两台虚拟机ssh的密码验证处于开启状态

6. 验证环境正常

```
1 # 在server和client任一虚拟机中 切换到 /vagrant/tju_tcp目录
2 vagrant@client:~$ cd /vagrant/tju_tcp/
3 # 使用make 编译初始代码
4 vagrant@client:/vagrant/tju_tcp$ make
5
6
7 # 在服务端运行服务端代码 5s内在客户端运行客户端代码
8 vagrant@server:/vagrant/tju_tcp$ /vagrant/tju_tcp/server
9 vagrant@client:/vagrant/tju_tcp$ /vagrant/tju_tcp/client
10 # 能够正常运行得到一些输出即为环境正常
```

7. 结束实验和继续实验

```
1 # 结束试验后, 通过vagrant halt命令方便地关闭两个虚拟机
2 vagrant halt
3 # 想继续进行实验时 使用vagrant up即可
4 vagrant up
```