VIRTUAL MACHINES

By Seth Lemanek

WHAT IS A VIRTUAL MACHINE?

- Software meant to emulate hardware for the purpose of hosting bare metal software like Operating Systems
- Used for creating virtual environments for safe and contained development, penetration testing, hosting cloud services, and more!
- Multiple VMs running on one host machine is managed by a hypervisor

WHAT TYPES OF VMS ARE THERE?

Type-1

- Runs directly on top of the machine
- Also called bare-metal hypervisor
- Includes:
  - Vmware ESX
  - Microsoft’s Hyper-V

Type-2

- Runs as a program from an operating system
- Includes:
  - Vmware WorkStation
  - Parallels Desktop
  - Oracle VirtualBox

SETUP TIME

- I have many USB Sticks that I will try to pass around as fast as I can.
- Some are USB 2.0, some are USB 3.0. If you have a USB 3.0, use a 3.0 stick for faster transfer speed.
- If you have one, copy over the VM_Workshop folder to your laptop.
- There will be VirtualBox installer and 2 VMs in a zip file
- Get VirtualBox installer running and extract the VMs

CREATING A VM

- Lets create a VM using
  - Click “New”
  - Type: Linux
  - Version: Ubuntu (64 bit)
  - Next, is the slider for your virtual RAM
    - If your PC is good, I recommend 2GB. If not, 1GB is fine
  - Creating a Virtual Disk File
    - Default is 10GB which is fine
    - If you wish to change later
  - Click Create

INSTALLING AN OS

- Your VM is ready but needs a bootable image to boot from (.iso)
- Settings > Storage > change virtual disk
- Browse to your image file and select
- Click start and your VM will boot of the image file as if it were a bootable CD
- Once finished installing, shut it down for the upcoming step.

ADDING AN EXISTING VM

- Once you copied the virtual machine folder, click the add button.
- Then browse to the folder called "Ubuntu Server" then click the Ubuntu Server.vbox file inside

SETTING UP A VIRTUAL NAT NETWORK

- By now, your VMs are running on Host-Only Networks
- File>Preferences>Network
- Click the plus symbol that says “adds new NAT network”
- Exit window then select a VM then settings
- Navigate to network, click dropdown box saying “NAT” and select “NAT Network” then click dropdown menu to select your new NAT Network
- Repeat for the other.

NAT NETWORKS

- Virtual Networks connect your VMs to the Internet through your host’s network connection
- Host-Only Networks: Separates each VM from each other but still connects them to the Internet if a connection is available
- NAT Networks: Similar to real NAT networks, it allows many VMS to share a network, gateway, and (maybe) DHCP servers.
- Bridged Networks: Bridges the VM(s) to the real network used by your host, meaning your VMs will use the same gateway and IP address space of your host.

TRYING OUT THE NETWORK

- Start both VMs
- Log into Ubuntu Server as “Server Administrator” with password “pass123”
- Log into your client with the credentials you made during installation
- Your VMs should have the different IP addresses under same network, test it with “ifconfig” in terminal
- In server’s terminal, type the following “cd Documents”, “nodejs server.js”
- In client open a web browser and type in address bar: “<serverip>:8000”
- You got a website! *Hopefully*

TAKING A SCREENSHOT (OR SNAPSHOT)

- Screenshot or snapshot saves amost everything about the VM’s state to a file.
- Useful for undoing mistakes or restoring your VM.
- To take on, Machine>Take Snapshot
- Give a unique name

TRYING OUT THE SNAPSHOT

- Oh my we have some junk JavaScript files from a test, let’s remove them.
- In terminal, type “rm *.js”.
- Oh no! We deleted all JavaScript files but we also deleted the main server code.
- To restore to your last snapshot, shut down the VM
- Click down arrow in “Machine Tools” and select “Snapshots”
- Select your snapshot, then click restore.
- Uncheck saving the current state then click restore.
- Start up the machine and it should be the same as well left it.

COPYING TO YOUR VM

- Select your VM
- Settings>General>Advanced
- Select dropdown menu for shared clipboard and drag-and-drop and select either “host-to-guest” or “bidirectional” for both.
- Start the machine then click machines>”insert the Guest Additions.iso”
- A CD appears on your machine, double click it and it should execute the “autorun.sh”
- Reboot and copy and paste works! Buuuut, only for copy and pasting text

ADDING A SHARED FOLDER

- This way uses shared folders between the host and the VM.
- Select Ubuntu Server then Settings>Shared Folders
- Click plus button to add new shared folder
- Navigate then select your folder
- Check on auto-mount not read-only
- Open your VM open terminal and type ”sudo adduser <username> vboxsf”
- You now have permission to read and write files from shared folder

COPYING YOUR ABOUT PAGE

- Copy and paste the about page to your new shared folder.
- Open up your shared folder in your Ubuntu server.
- Copy and paste from folder to the Documents folder
- Then run server.js

THE END