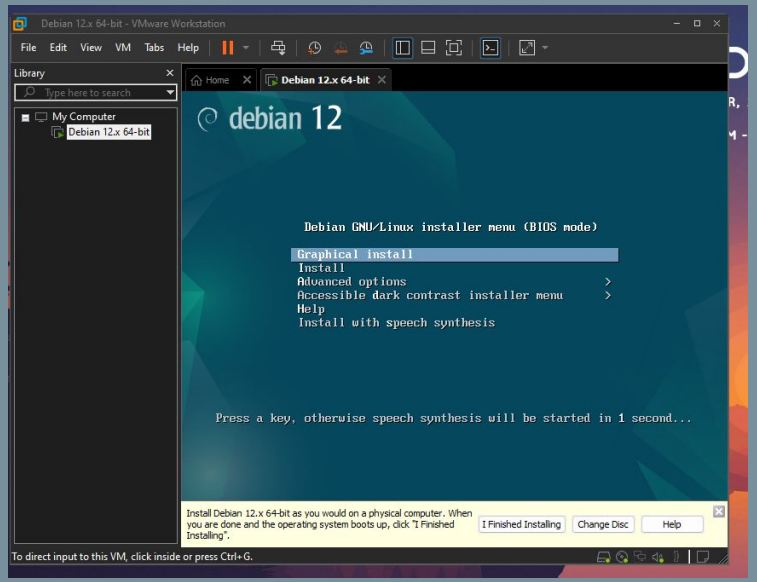
Final Project/Portfolio piece

By: Ashton Rowe

Creating Debian 12 VM



This was pretty simple, I had problems with connecting the vm to the internet but got it all figured out. I did do the graphical install first but then when i was logged in, I put a command in to not have a GUI

Adding Openssh server and webserver

First i had to Install the OpenSSHserver to allow access. Then i had to Start the SSH service so it's ready to use, after that then i would Enable it to run automatically every time the system boots. I had to Find the computer's IP address to connect to it. Connect to a webserver like Apache to host a website then start the webserver. Not too hard

Debian 12.x 64-bit (9) - VMware Workstation

File Edit View VM Tabs Help

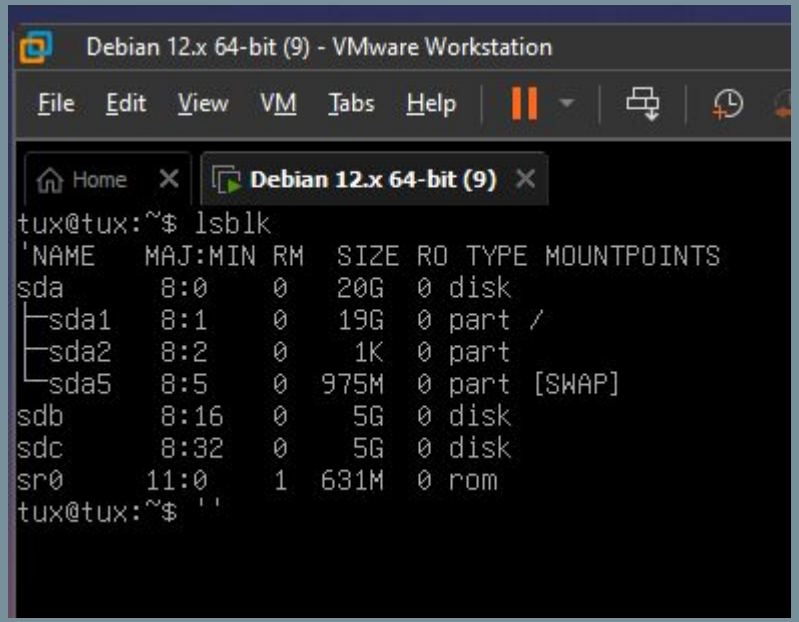
Home | Debian 12.x 64-bit (9)

```
tux@tux:~$ apt update
Reading package lists... Done
E: Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)
E: Unable to lock directory /var/lib/apt/lists/
W: Problem unlinking the file /var/cache/apt/pkgcache.bin - RemoveCaches (13: Permission denied)
W: Problem unlinking the file /var/cache/apt/srcpkgcache.bin - RemoveCaches (13: Permission denied)
tux@tux:~$ apt install openssh-server
E: Could not open lock file /var/lib/dpkg/lock-frontend - open (13: Permission denied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), are you root?
tux@tux:~$ su -
Password:
root@tux:~# apt update
Hit:1 http://security.debian.org/debian-security bookworm-security InRelease
Hit:2 http://deb.debian.org/debian bookworm InRelease
Hit:3 http://deb.debian.org/debian bookworm-updates InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
root@tux:~# apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssh-server is already the newest version (1:9.2p1-2+deb12u3).
openssh-server set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@tux:~# exit
logout
tux@tux:~$ systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-12-09 21:12:09 PST; 5min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 752 (sshd)
      Tasks: 1 (limit: 2243)
     Memory: 3.3M
        CPU: 18ms
     CGroup: /system.slice/ssh.service
             └─752 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Warning: some journal files were not opened due to insufficient permissions.
tux@tux:~$
```

```
tux@tux:~$ systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-12-10 15:19:59 PST; 29min ago
       Docs: https://httpd.apache.org/docs/2.4/
   Main PID: 913 (apache2)
      Tasks: 56 (limit: 2238)
     Memory: 16.3M
        CPU: 161ms
     CGroup: /system.slice/apache2.service
             ├─913 /usr/sbin/apache2 -k start
             ├─914 /usr/sbin/apache2 -k start
             ├─915 /usr/sbin/apache2 -k start
             └─916 /usr/sbin/apache2 -k start

Warning: some journal files were not opened due to insufficient permissions.
tux@tux:~$
```

To direct input to this VM, click inside or press Ctrl+G.

Adding 2 small disk



I went into the Vm workstation settings then added two 5 GB hard disk to the new VM, then powered on the VM and put in Lsblk to confirm that the hard disk that i put in are there. I used VM workstation

- ***creating 1 volume group (vg)***
- ***creating 1 mount point (lv)***
- ***Adding mount point to /etc/fstab using its UUID from blkid***

```
tux@tux:~$ lsblk
NAME                MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda                   8:0    0   20G  0 disk
├─sda1                8:1    0   19G  0 part /
├─sda2                8:2    0    1K  0 part
└─sda5                8:5    0  975M  0 part [SWAP]
sdb                   8:16   0    5G  0 disk
└─vg_data-lv_data 254:0    0    9G  0 lvm  /mnt/data
sdc                   8:32   0    5G  0 disk
└─vg_data-lv_data 254:0    0    9G  0 lvm  /mnt/data
sr0                  11:0    1  631M  0 rom
```

```
sdc      8:32   0    5G  0 disk
sr0     11:0    1  631M  0 rom
tux@tux:~$ sudo apt install -y lvm2
[sudo] password for tux:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lvm2 is already the newest version (2.03.16-2).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
tux@tux:~$ sudo pvcreate /dev/sdX /dev/sdY
  No device found for /dev/sdX.
  No device found for /dev/sdY.
tux@tux:~$ sudo pvcreate /dev/sdc /dev/sdb
  Physical volume "/dev/sdc" successfully created.
  Physical volume "/dev/sdb" successfully created.
tux@tux:~$ sudo vgcreate vg_data /dev/sdc /dev/sdb
  Volume group "vg_data" successfully created
tux@tux:~$ sudo lvcreate -L 9G -n lv_data vg_data
  Logical volume "lv_data" created.
tux@tux:~$ sudo mkfs.ext4 /dev/vg_data/lv_data
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 2359296 4k blocks and 589824 inodes
Filesystem UUID: 984cb447-1b42-4f99-b213-d77627fb667c
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

tux@tux:~$ sudo mkdir /mnt/data
tux@tux:~$ sudo mount /dev/vg_data/lv_data /mnt/data
tux@tux:~$ _
```

Explaining Last slide

So when i went to create a volume group and a logical volume , I had to add the mount point to `/etc/fstab` . I also had to make the disks ready by initializing them with `pvcreate`, and then created a volume group using those disks with `vgcreate`. Then i created a logical volume within that volume group with `lvcreate`, specifying the size. When the logical volume was created, i had to format it with a filesystem using `mkfs.ext4` and mounted it to a directory (e.g., `/mnt/data`) I used the `blkid` command to get the UUID of the logical volume and then added an entry to `/etc/fstab` with the UUID to automatically mount it to the specified directory.

Adding tux user and sudo for tux

```
tux@tux:~$ sudo ls /
bin   dev  home        initrd.img.old  lib64       media  opt   root  sbin  sys  usr  vmlinuz
boot  etc  initrd.img  lib             lost+found  mnt    proc  run   srv   tmp  var  vmlinuz.old
tux@tux:~$
```

```
tux@tux:~$ id tux
uid=1000(tux) gid=1000(tux) groups=1000(tux),24(cdrom),25(floppy),27(sudo),29(audio),30(dip),44(video),46(plugdev),100(users),106(netdev),111(bluetooth),113(lpa
dmin),116(scanner)
tux@tux:~$
```

This was quite simple. I did this first so i could use sudo command when logged in with tux. First i had to go into root using the su - command. Then i added a user then used sudo usermod -aG sudo tux to add tux to the sudo group. The two pictures show that tux user is made and that tux is added to the user group

Setting up Serial Console

```
tux@tux:~$ GRUB_CMDLINE_LINUX="console=tty50"
tux@tux:~$ sudo update-grub
Generating grub configuration file ...
Found background image: /usr/share/images/desktop-base/desktop-grub.png
Found linux image: /boot/vmlinuz-6.1.0-28-amd64
Found initrd image: /boot/initrd.img-6.1.0-28-amd64
Found linux image: /boot/vmlinuz-6.1.0-27-amd64
Found initrd image: /boot/initrd.img-6.1.0-27-amd64
Warning: os-prober will not be executed to detect other bootable partitions.
Systems on them will not be added to the GRUB boot configuration.
Check GRUB_DISABLE_OS_PROBER documentation entry.
done
tux@tux:~$ sudo systemctl enable serial-getty@tty50.service
Created symlink /etc/systemd/system/getty.target.wants/serial-getty@tty50.service → /lib/systemd/system/serial-getty@.service.
tux@tux:~$ _
```

When setting up the serial console, I enabled the serial console service by ensuring that the system would use the serial port for login access. I did this by going in and configuring `systemd` to start a getty service on the serial port which would allow me log in through the serial console. I modified the system configuration by creating or editing files to ensure the serial console would start on boot and accept login attempts. After all that i tested the setup by connecting to the serial console.

- ***create 2 new directories bin and inclass for the tux user and adding them to $PATH***

```
tux@tux:~$ mkdir ~/bin ~/inclass
tux@tux:~$ echo 'export PATH=$PATH:~/bin:~/inclass' >> ~/.bashrc
tux@tux:~$ source ~/.bashrc
tux@tux:~$
tux@tux:~$ echo $PATH
/home/tux/bin:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/home/tux/bin:/home/tux/inclass
tux@tux:~$
```

So I created two new directories, `bin` and `inclass` for tux and add them to the $PATH, I created the directories inside the `tux` user's home directory using the `mkdir` command. Then, I updated the `tux` user's environment by modifying the `.bashrc` file. In this file, I added a line to include the newly created directories which is (`~/bin` and `~/inclass`) in the $PATH variable, allowing me to run scripts or programs stored in those directories from anywhere in the terminal. After making this change,I used `source ~/.bashrc` to apply the update.

Adding .funcs

```
tux@tux:~$ calcit() { echo "$1" | bc; }
tux@tux:~$ whdr() { curl -sI "$1" | head -n 1; }
tux@tux:~$ wtfr() { curl -s "$1"; }
tux@tux:~$ echo "source ~/.funcs" >> ~/.bashrc
tux@tux:~$ source ~/.funcs
-bash: /home/tux/.funcs: No such file or directory
tux@tux:~$ calcit "5 * 5"
25
```

When adding the functions for `tux`, I created a file called `.funcs` in the user's home directory. In this file, I defined custom functions like `calcit`, `whdr` and `wftr`. After defining the functions in `.funcs`, I updated the `.bashrc` file to automatically load the functions by adding a line that sources the `.funcs` file. The I used the `source ~/.bashrc` command to apply the changes, making the functions available.