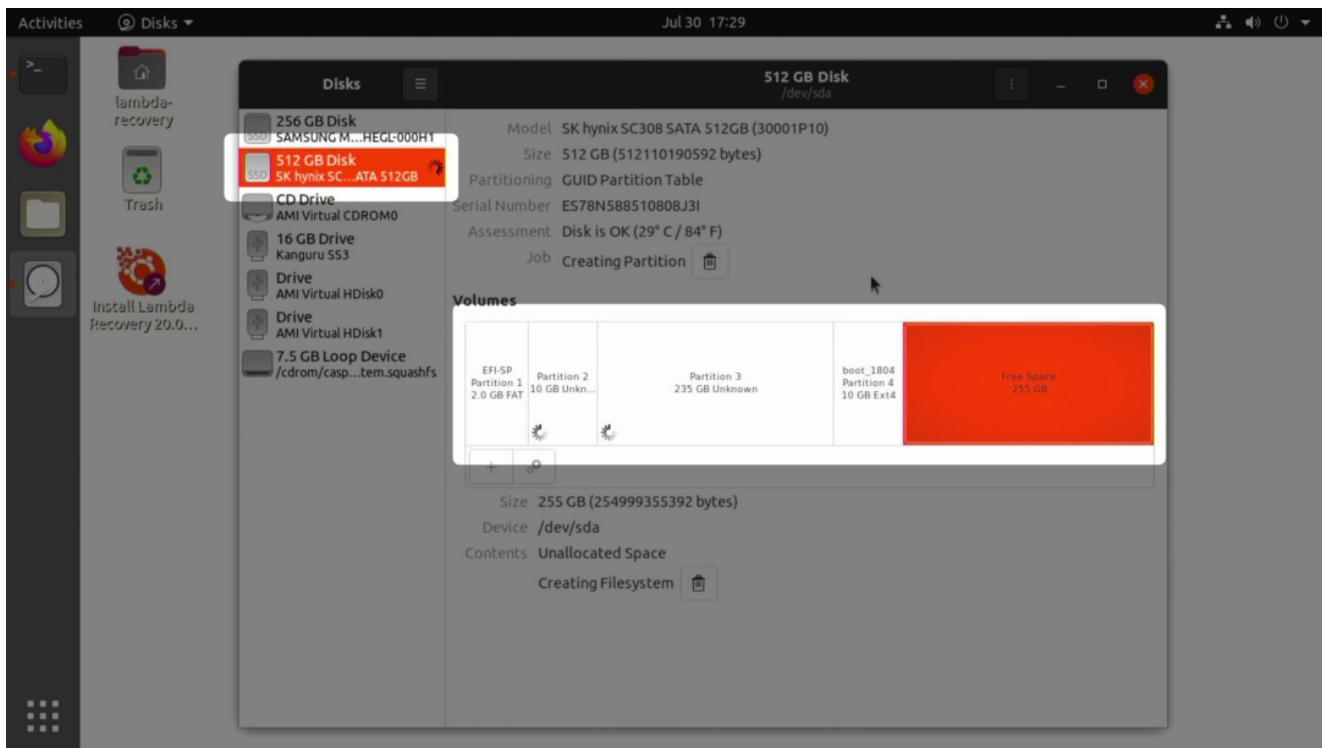


Dual Boot 20.04 and 18.04 with LUKS

1



Format the SATA device and create 5 new partitions

Part1 will be the EFI boot partition

Part2 will be the 20.04 /boot partition

Part3 will be the 20.04 LUKS partition

Part4 will be the 18.04 /boot partition

Part5 will be the 18.04 LUKS partition

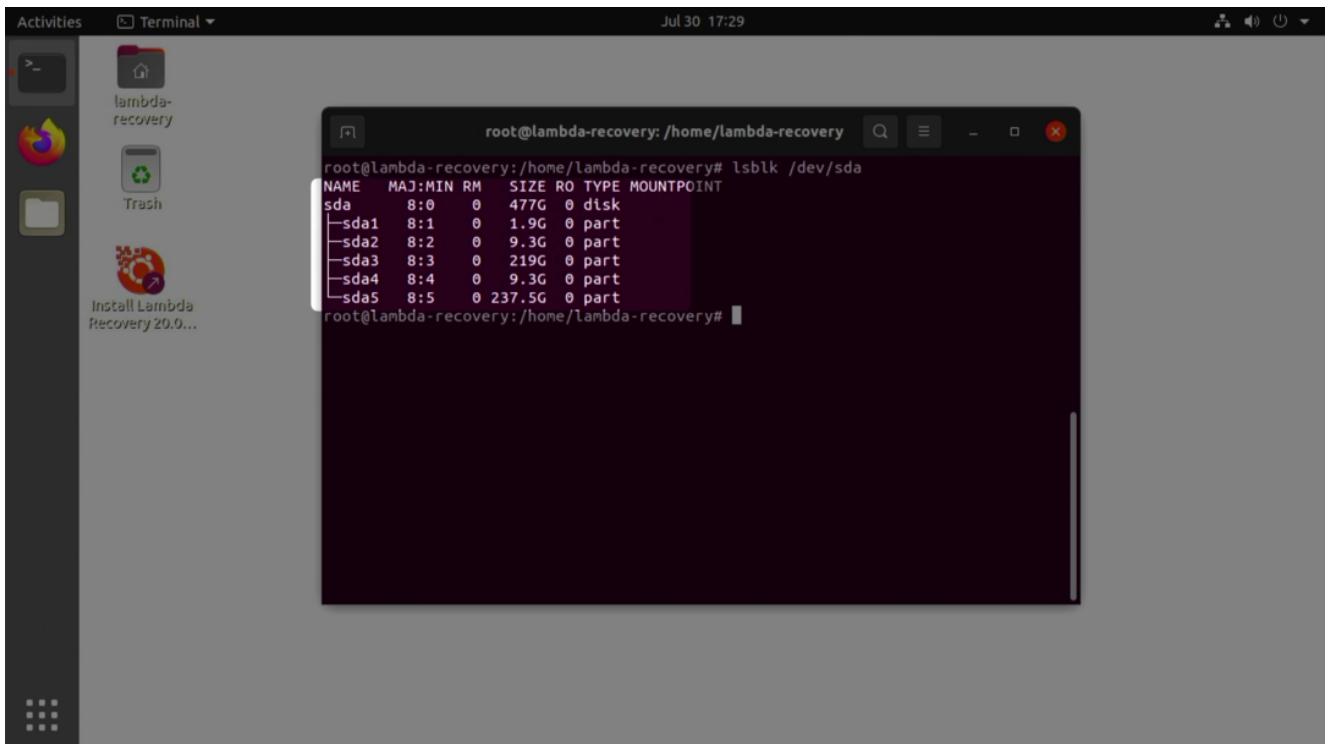
Partition 1 should be 2GB

The boot partitions should be big enough to hold all of the kernels that will ever be installed 10GB is very safe

The LUKS partitions should half of the total drive capacity each, less the 2GB used for EFI and the amount used for the boot partitions.

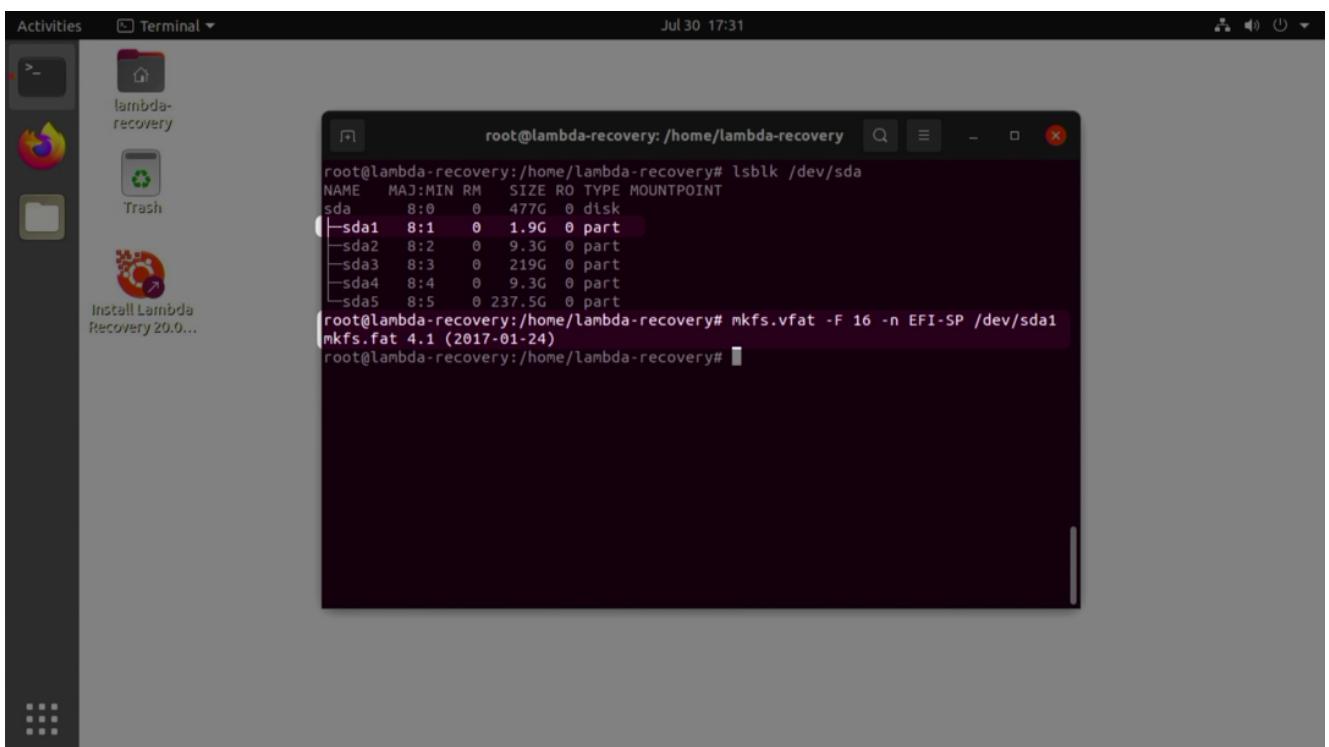
NOTE-- These will all be formatted later on. DO NOT select the encryption options. Use defaults for now

2



This guide assumes that the drive is labeled as SDA. This is what the output of `lsblk /dev/sda` should look like

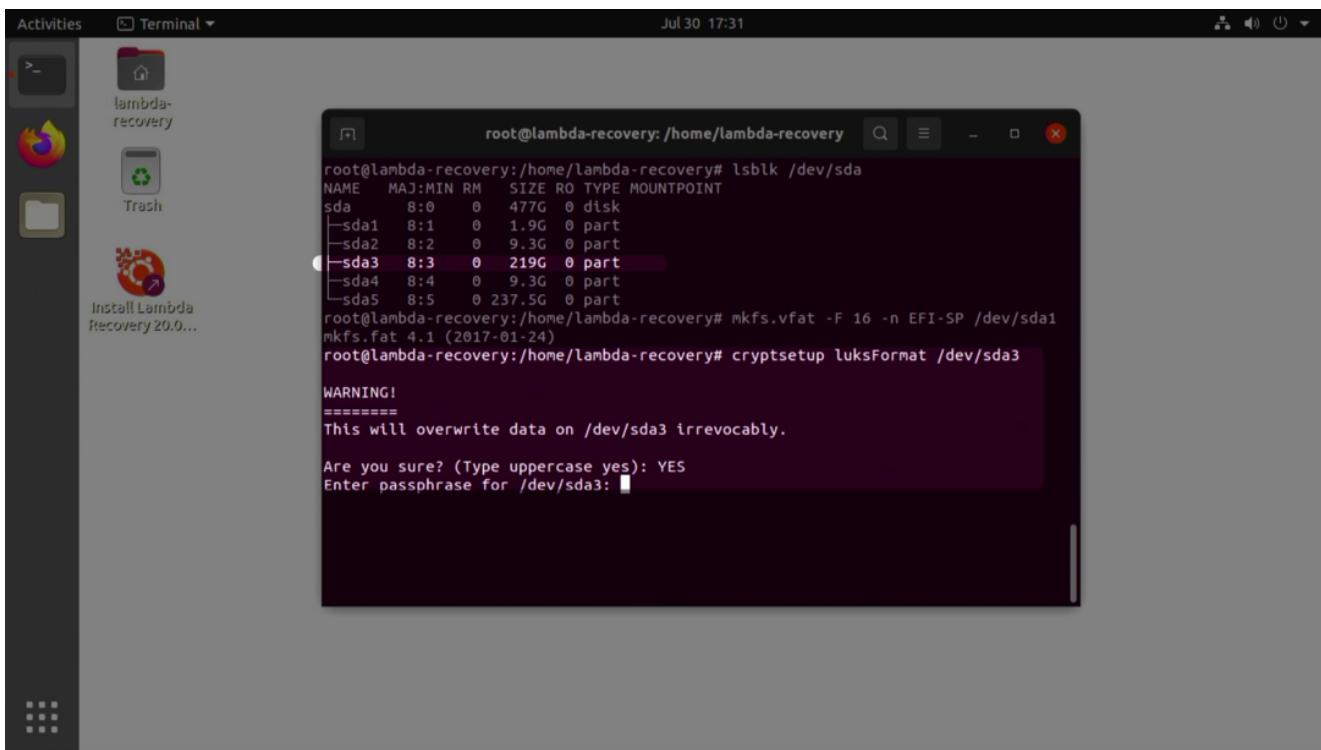
3



First we will setup /dev/sda1 for use as the EFI partition

```
mkfs.vfat -F 16 -n EFI-SP /dev/sda1
```

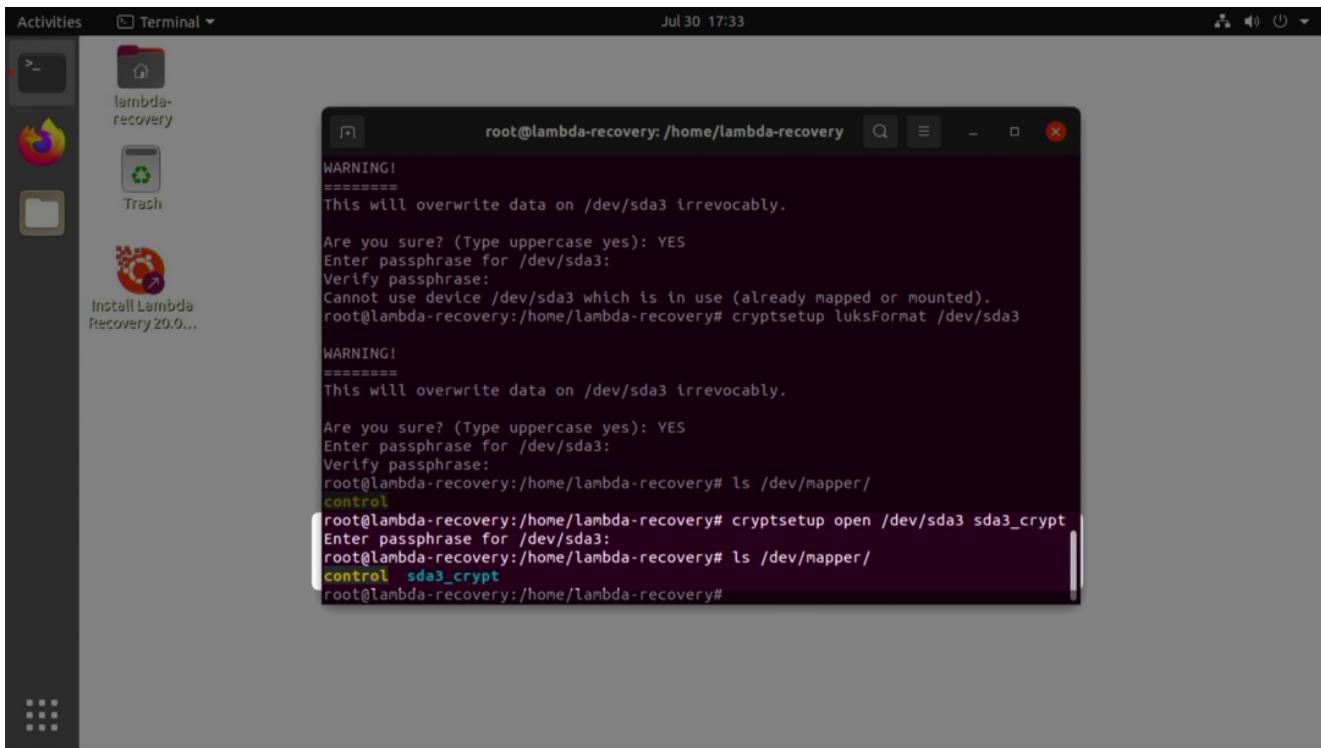
4



Now we will build the LUKS container for the 20.04 root partition

```
cryptsetup luksFormat /dev/sda3
```

5



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "root@lambda-recovery: /home/lambda-recovery". The terminal displays the following command sequence:

```
WARNING!
=====
This will overwrite data on /dev/sda3 irrevocably.

Are you sure? (Type uppercase yes): YES
Enter passphrase for /dev/sda3:
Verify passphrase:
root@lambda-recovery:/home/lambda-recovery# cryptsetup luksFormat /dev/sda3

WARNING!
=====
This will overwrite data on /dev/sda3 irrevocably.

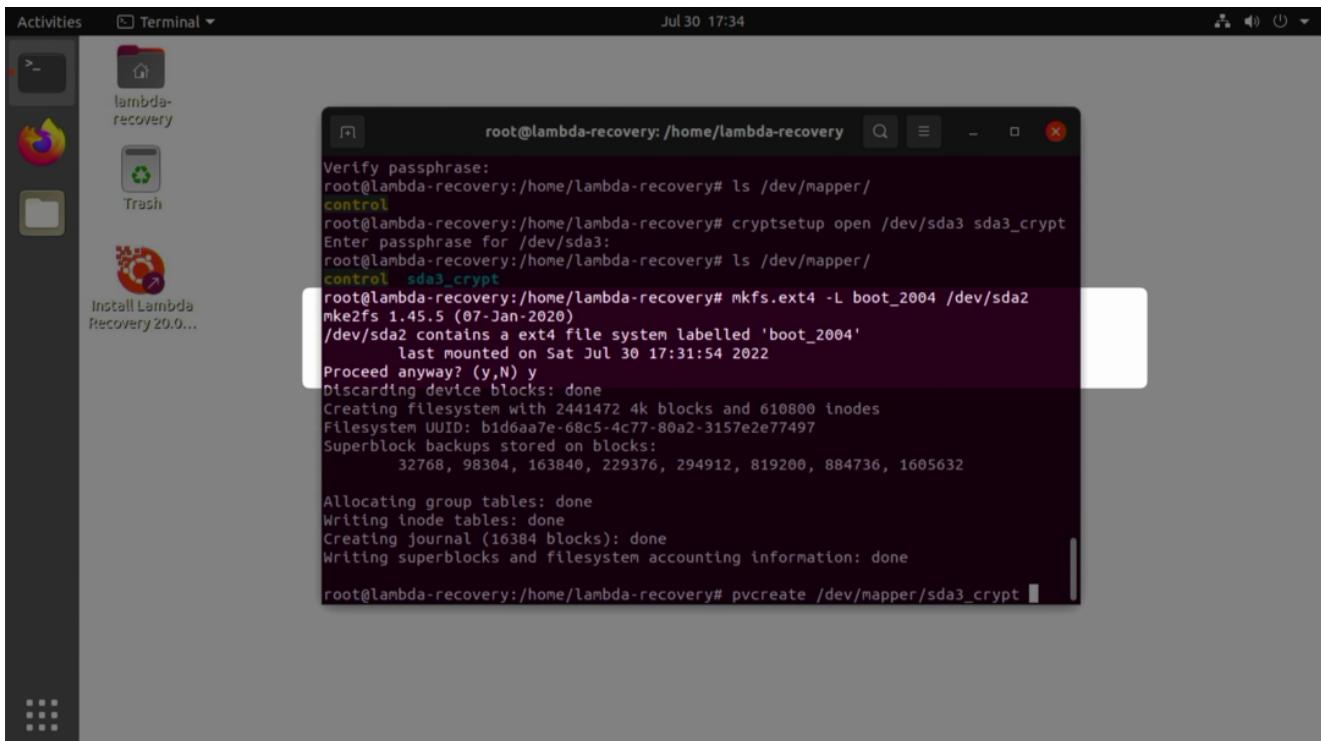
Are you sure? (Type uppercase yes): YES
Enter passphrase for /dev/sda3:
Verify passphrase:
root@lambda-recovery:/home/lambda-recovery# ls /dev/mapper/
control
root@lambda-recovery:/home/lambda-recovery# cryptsetup open /dev/sda3 sda3_crypt
Enter passphrase for /dev/sda3:
root@lambda-recovery:/home/lambda-recovery# ls /dev/mapper/
control sda3_crypt
root@lambda-recovery:/home/lambda-recovery#
```

Now we will decrypt and map /dev/sda3 to sda3_crypt

```
cryptsetup open /dev/sda3 sda3_crypt
```

```
ls /dev/mapper shows that is is mapped
```

6



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "root@lambda-recovery: /home/lambda-recovery". The terminal displays the following command sequence:

```
Verify passphrase:
root@lambda-recovery:/home/lambda-recovery# ls /dev/mapper/
control
root@lambda-recovery:/home/lambda-recovery# cryptsetup open /dev/sda3 sda3_crypt
Enter passphrase for /dev/sda3:
root@lambda-recovery:/home/lambda-recovery# ls /dev/mapper/
control sda3_crypt
root@lambda-recovery:/home/lambda-recovery# mkfs.ext4 -L boot_2004 /dev/sda2
mke2fs 1.45.5 (07-Jan-2020)
/dev/sda2 contains a ext4 file system labelled 'boot_2004'
last mounted on Sat Jul 30 17:31:54 2022
Proceed anyway? (y,N) y
Discarding device blocks: done
Creating filesystem with 2441472 4k blocks and 610800 inodes
Filesystem UUID: b1d6aa7e-68c5-4c77-80a2-3157e2e77497
Superblock backups stored on blocks:
      32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

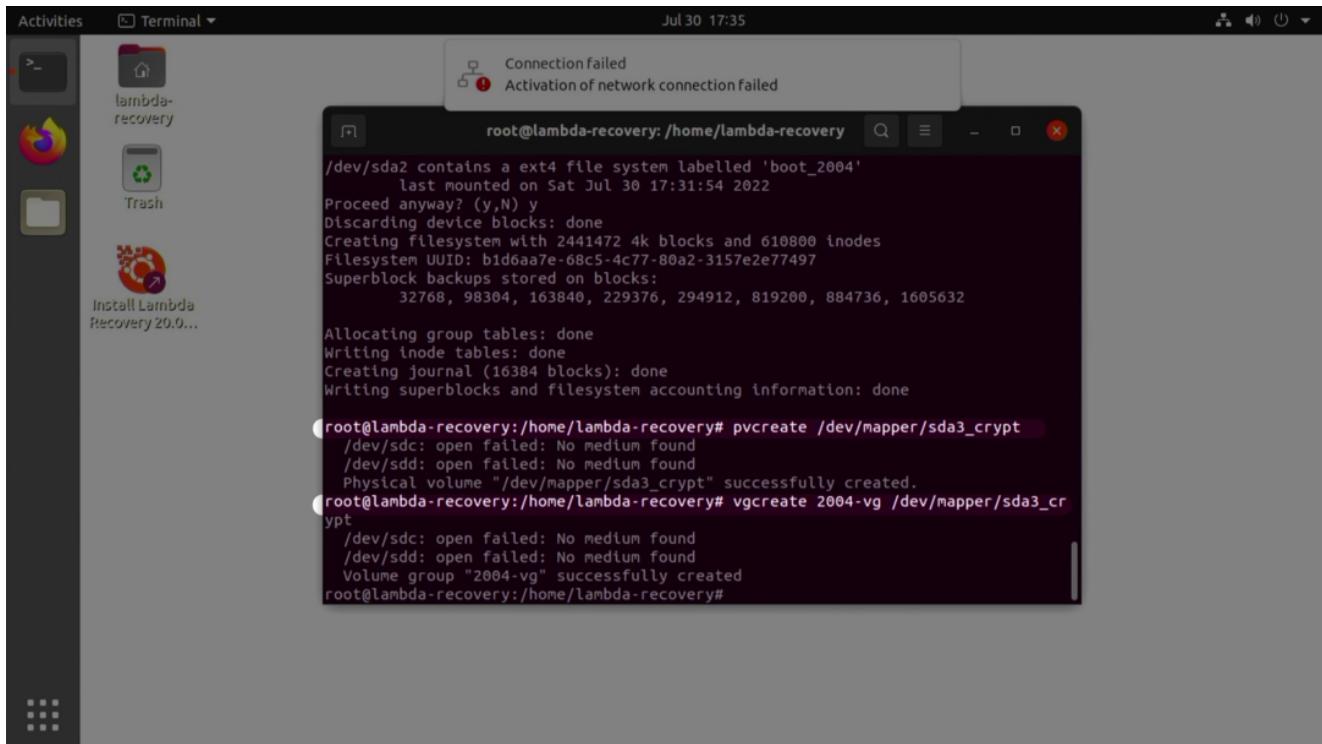
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

root@lambda-recovery:/home/lambda-recovery# pvcreate /dev/mapper/sda3_crypt
```

Format /dev/sda2 as ext4

```
mkfs.ext4 L boot_2004 /dev/sda2
```

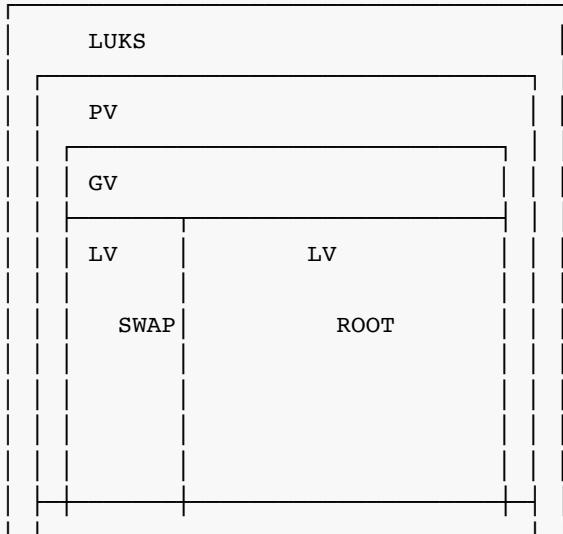
7



We will use LVM under the LUKS partition. This cleared up the major issues I was having in other testing and all of the documentation pointed to use LVM

```
pvcreate /dev/mapper/sda3_crypt  
vgcreate 2004-vg /dev/mapper/sda3_crypt  
lvcreate -L 4G -n swap_1 2004_vg  
lvcreate -l 80%FREE -n root 2004-vg
```

Block diagram of LUKS and LVM

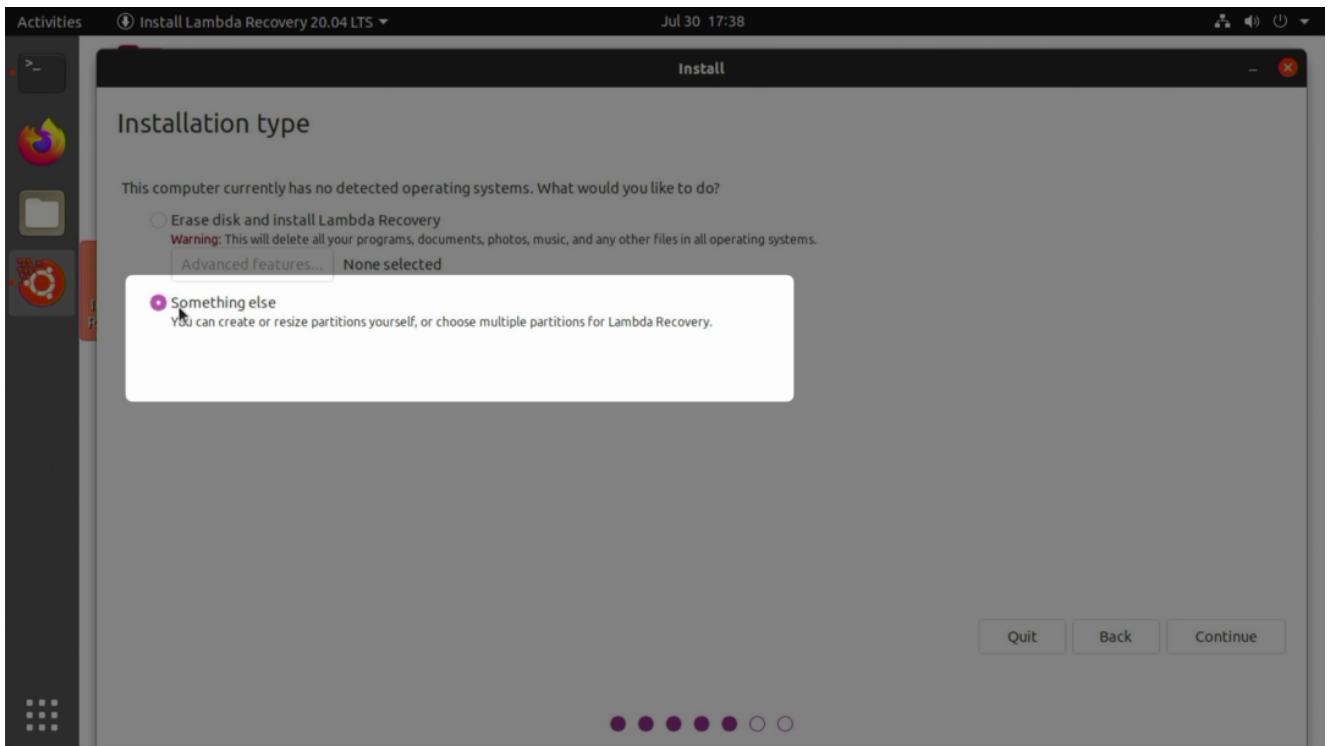


8

A screenshot of a Linux desktop environment showing a terminal window. The terminal window title is "root@lambda-recovery:/home/lambda-recovery". The terminal content shows the following LVM commands and their outputs:

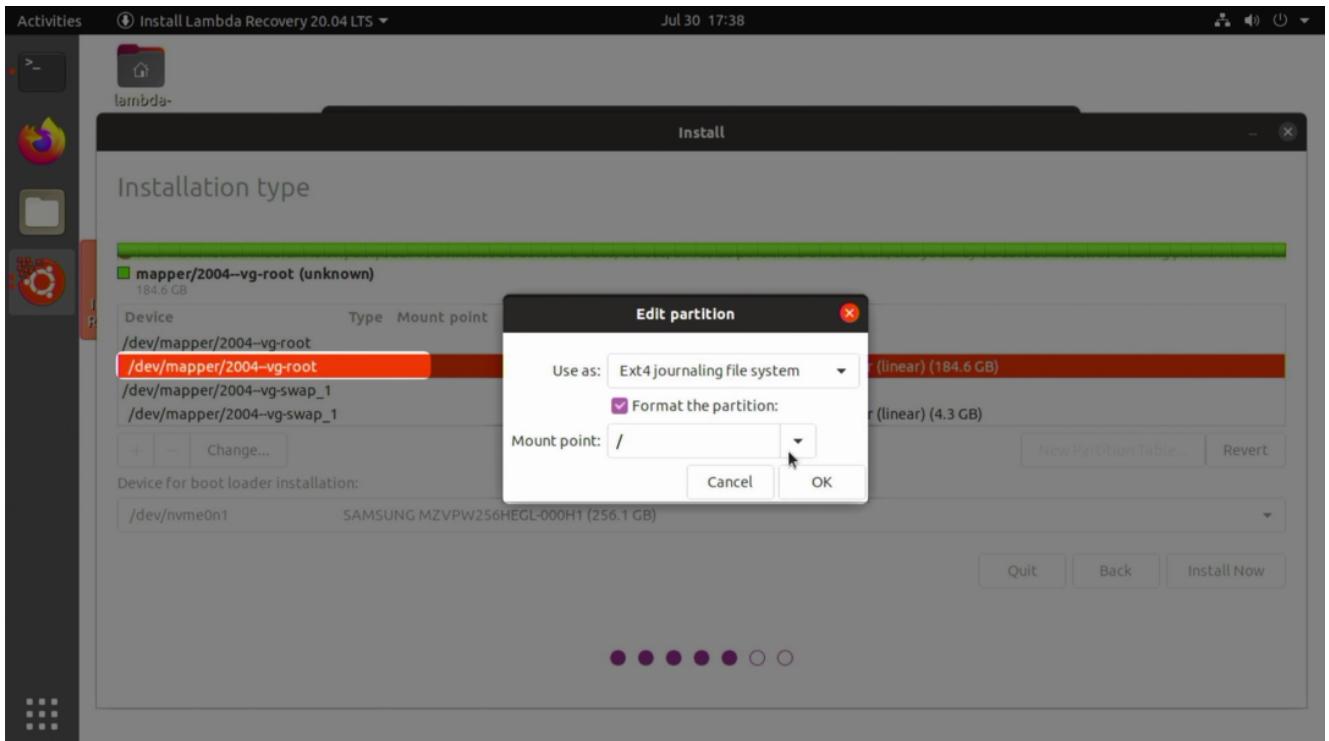
```
Volume group "2004-vg" successfully created
root@lambda-recovery:/home/lambda-recovery# lvcreate -L 4G -n swap_1 ubuntu-vg
/dev/sdc: open failed: No medium found
/dev/sdd: open failed: No medium found
Volume group "ubuntu-vg" not found
Cannot process volume group ubuntu-vg
root@lambda-recovery:/home/lambda-recovery# vg
vgcfgbackup    vgconvert    vgexport    vgmerge      vgrename
vgcfgrestore   vgcreate     vgextend     vgknodes    vgs
vgchange       vgdb         vgimport    vgreduce    vgscan
vgck          vgdisplay   vgimportclone vgremove   vgsplit
root@lambda-recovery:/home/lambda-recovery# vgs
/dev/sdc: open failed: No medium found
/dev/sdd: open failed: No medium found
Found volume group "2004-vg" using metadata type lvm2
root@lambda-recovery:/home/lambda-recovery# lvcreate -L 4G -n swap_1 2004-vg
/dev/sdc: open failed: No medium found
/dev/sdd: open failed: No medium found
Logical volume "swap_1" created.
root@lambda-recovery:/home/lambda-recovery# lvcreate -l 80%FREE -n root 2004-vg
```

9



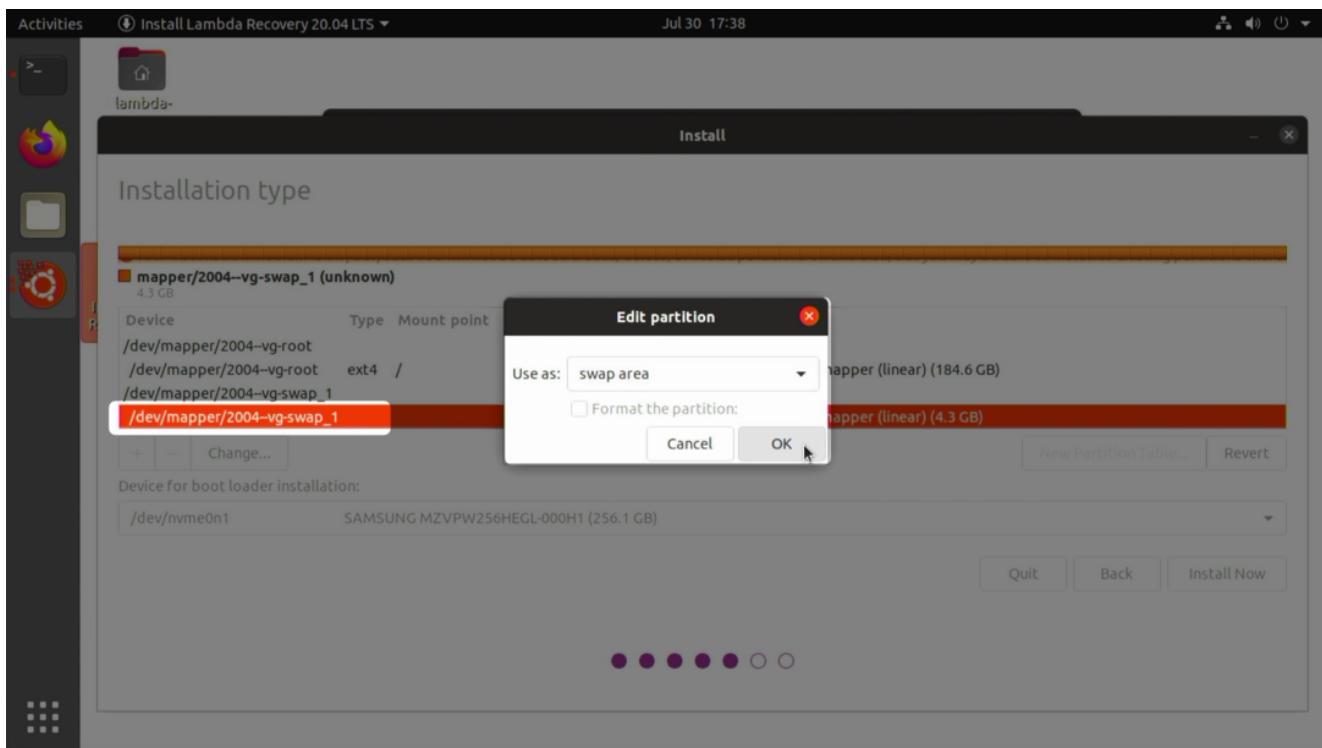
Launch the Installer and choose 'Something else' as the Installation type

10



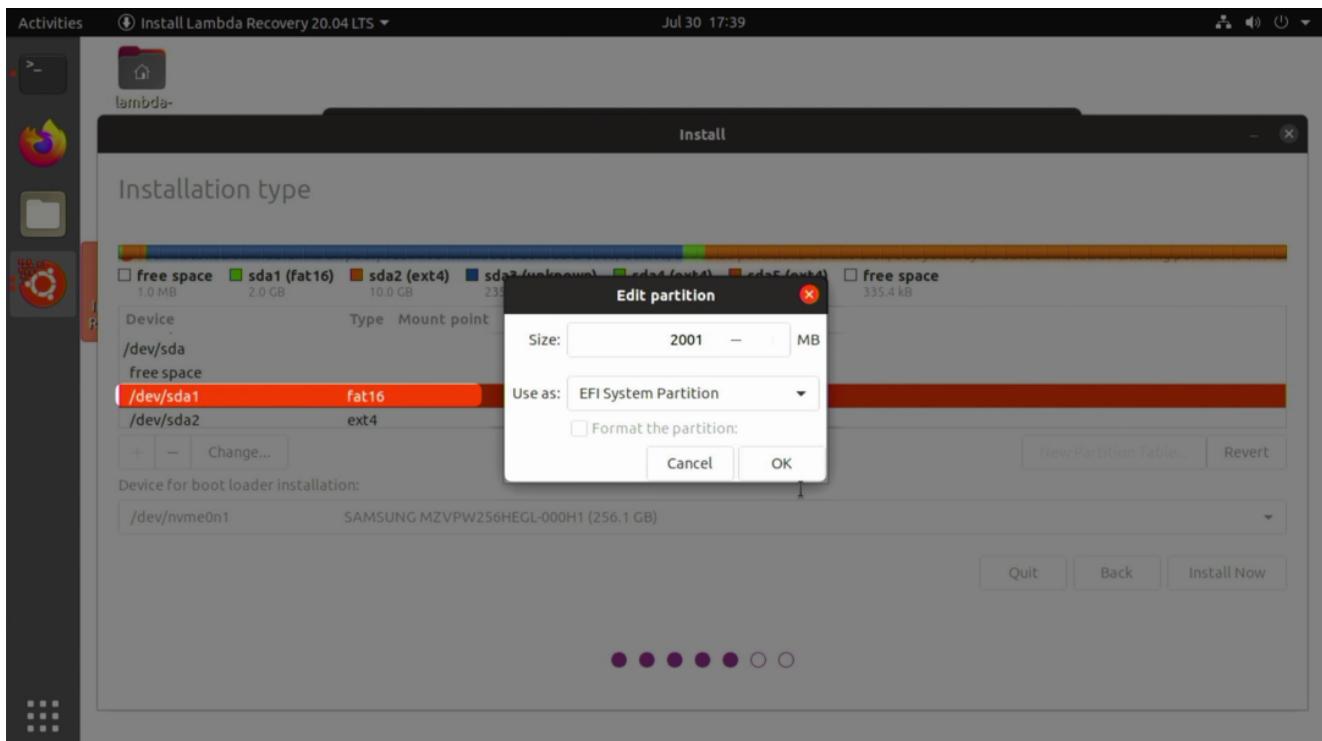
Select the child /dev/mapper/2004-vg-root and edit the partition as EXT4, format the partition, and set '/' as the mount point

11



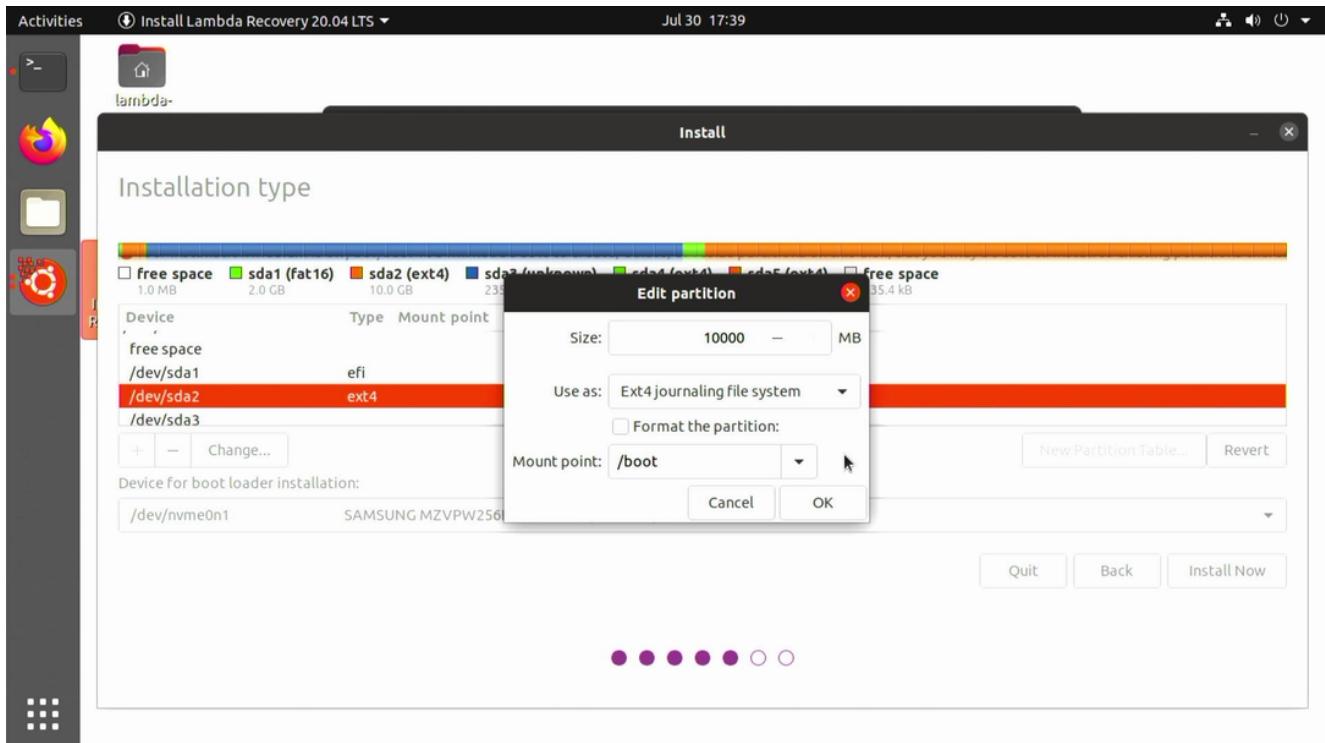
Select the child /dev/mapper/2004-vg-swap_1 and 'Use as: swap area'

12



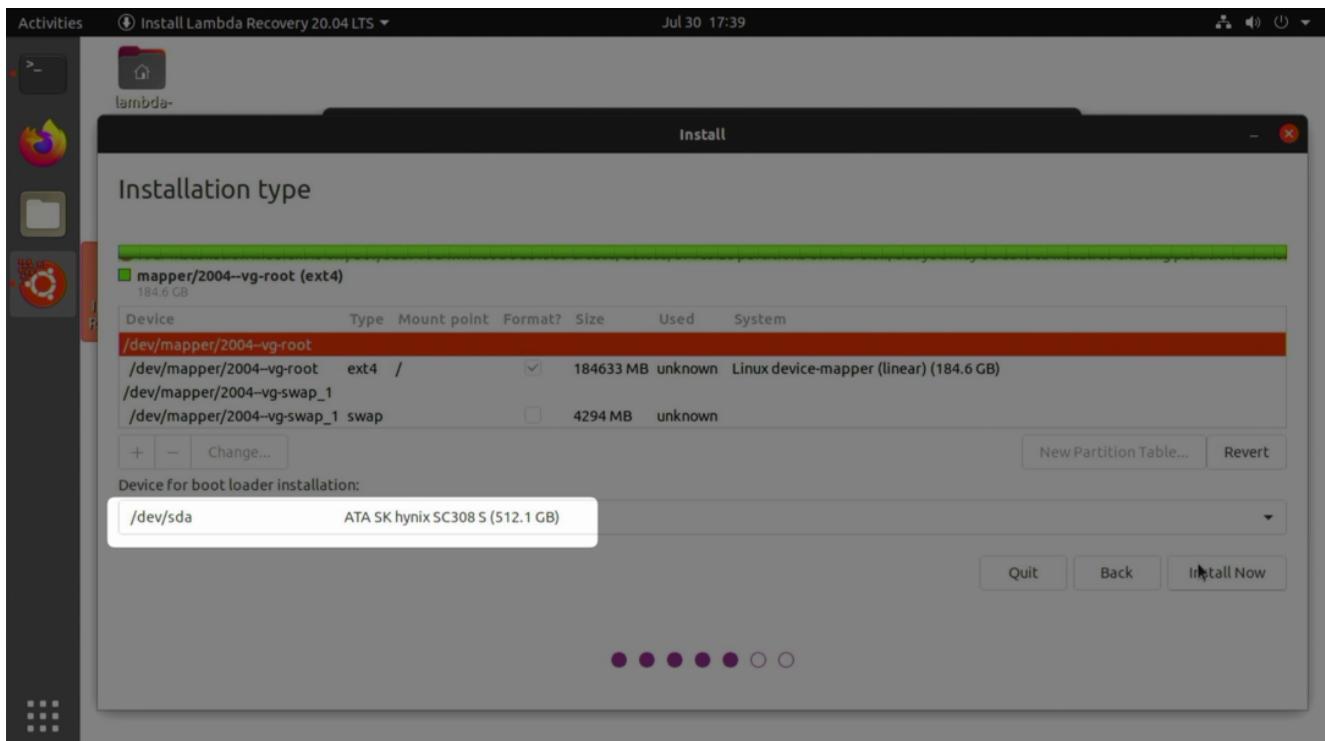
Select /dev/sda1 and 'Use as: EFI System Partition'

13



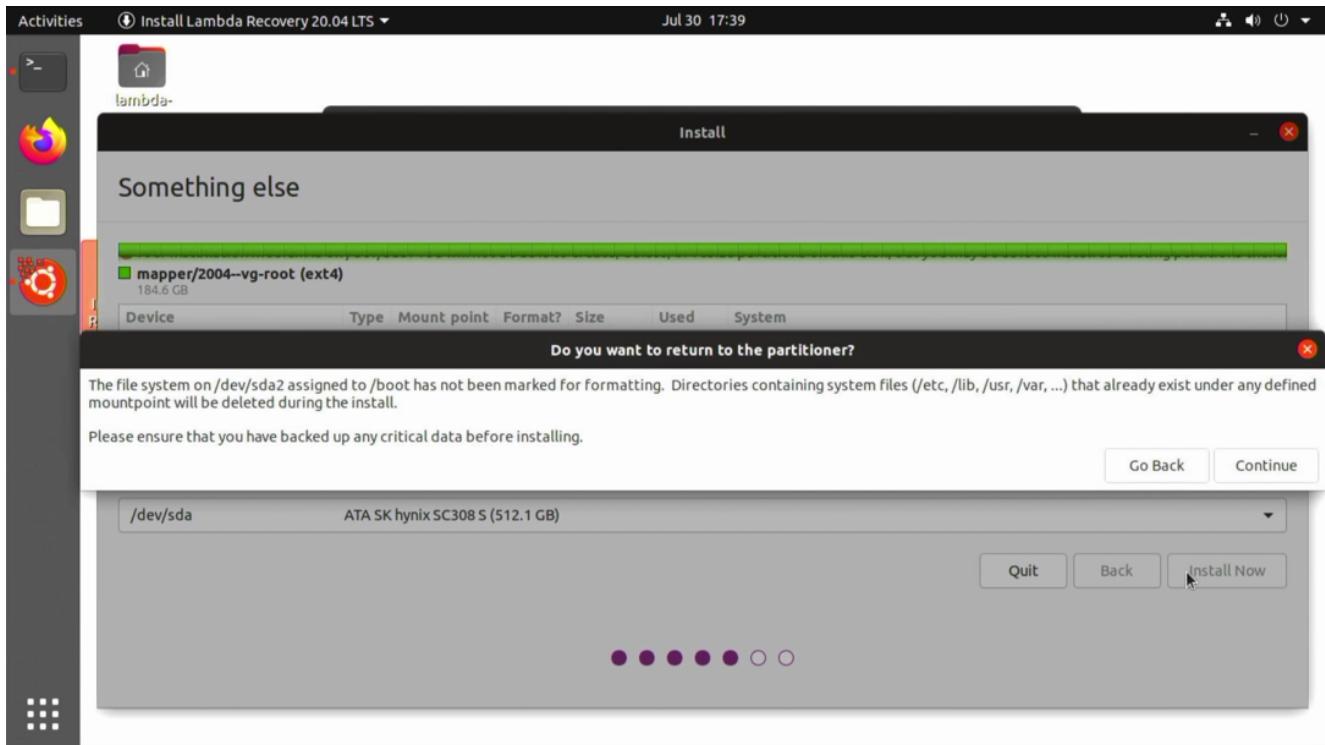
Select /dev/sda2 and Format as EXT4, format the partition and set mount point as '/boot'

14

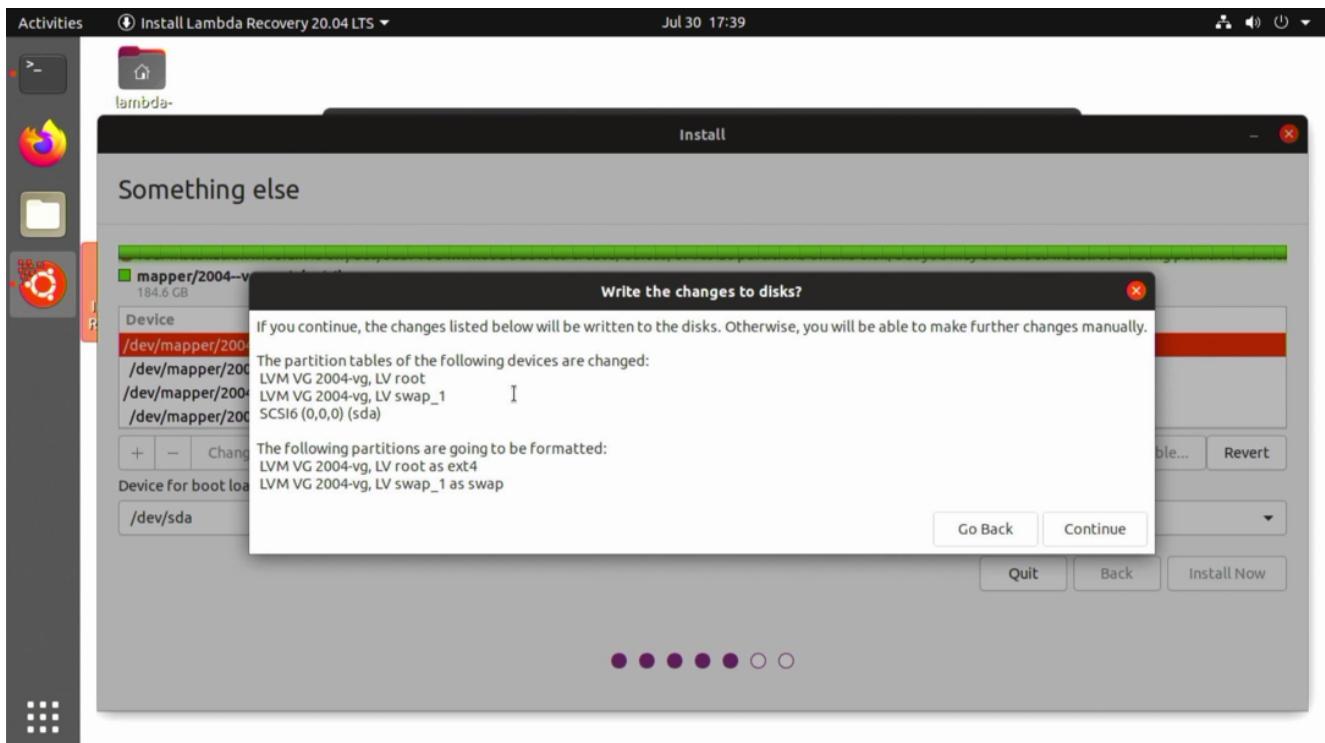


Select the root device-- /dev/sda as the bootloader installation device

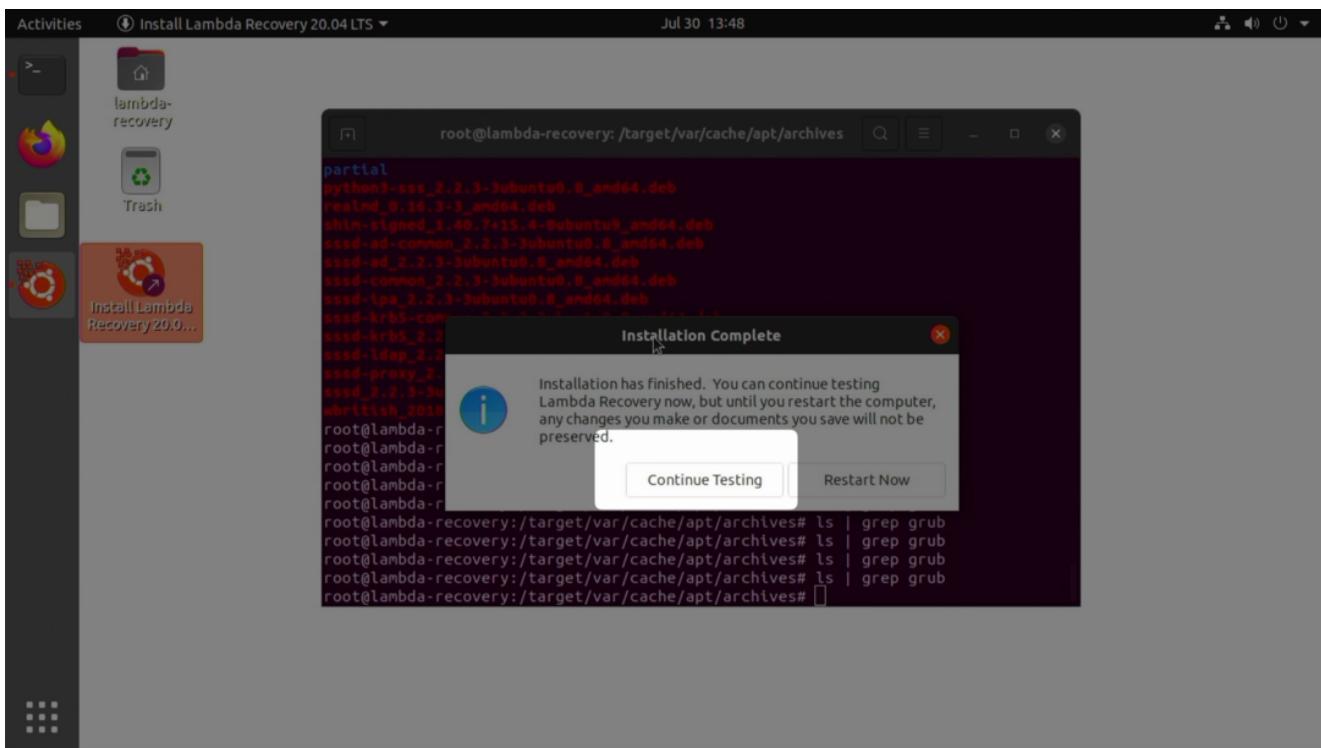
15



16

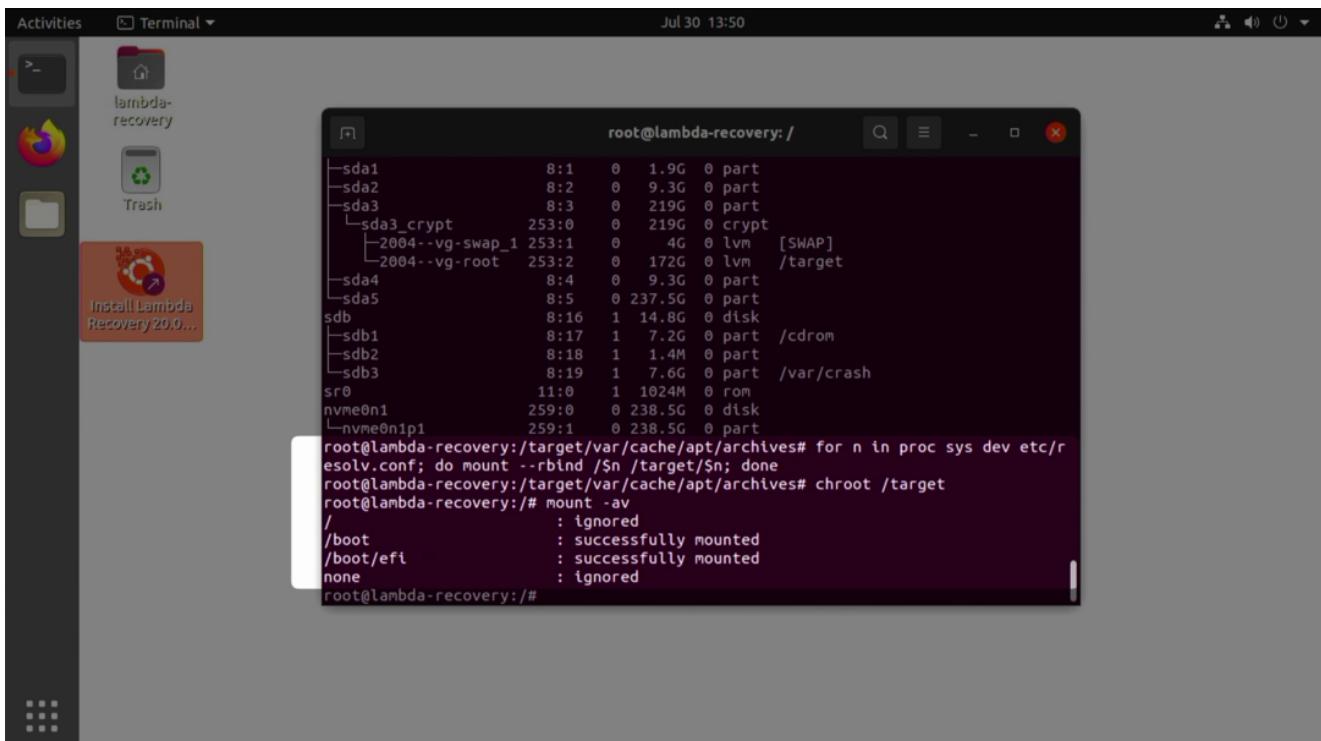


17



After Installation is complete, choose Continue Testing

18



We need to chroot into the new install so we have to mount the proc sys dev and etc/resolv devices onto the new

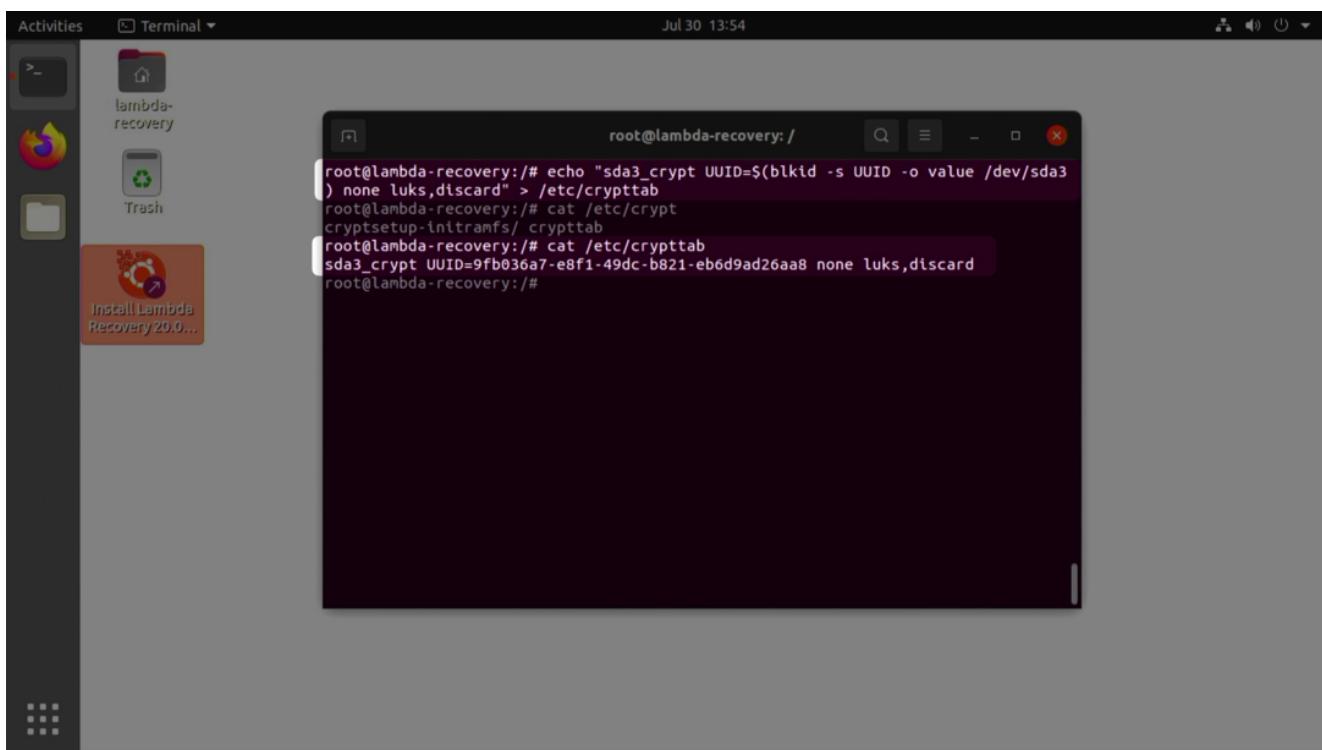
install before we reboot

```
for n in proc sys dev etc/resolv.conf; do mount --rbind /$n /target/$n; done  
chroot /target  
mount -av
```

you should see:

```
/  
/boot  
/boot/efi
```

19



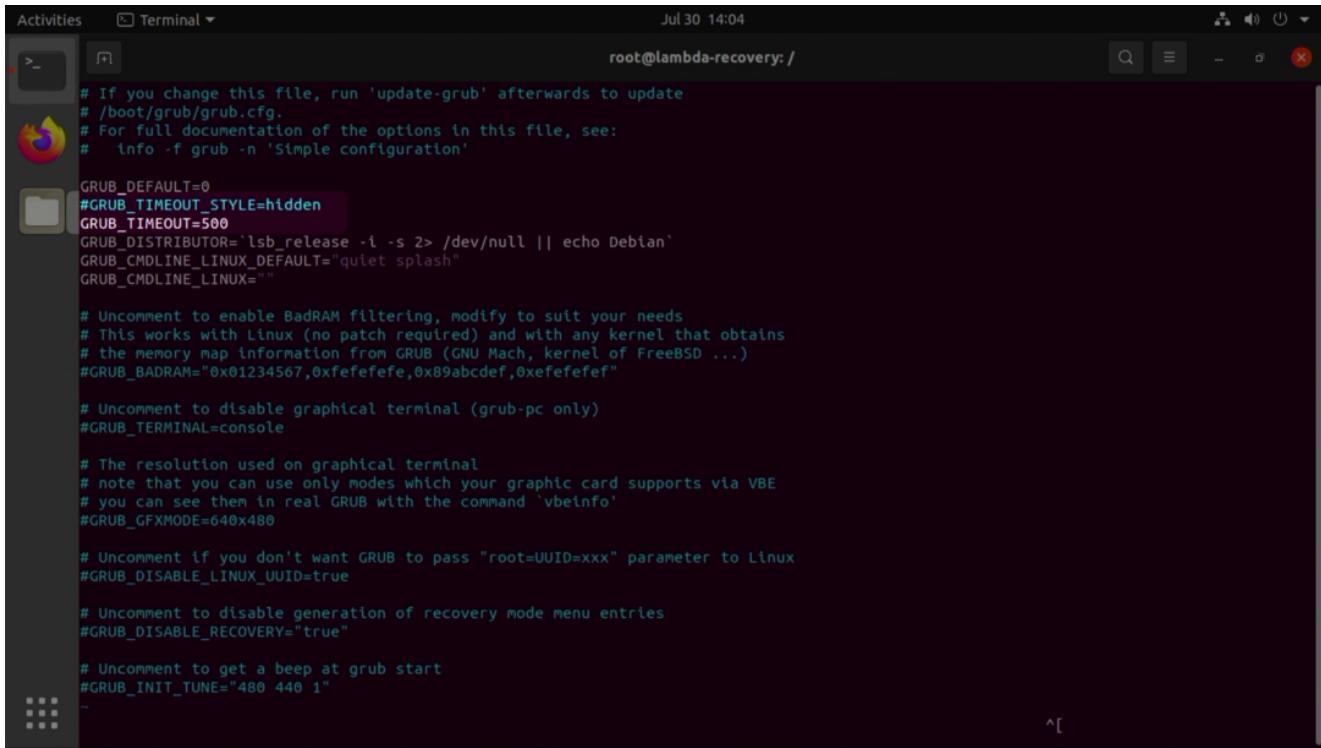
Create a crypttab file

sda3_crypt is the mapper label for sda3, the UUID should be for /dev/sda3 and 'none luks,discard' options are added

This one-liner will set it up correctly, assuming again that the device is /dev/sda

```
echo "sda3_crypt UUID=$(blkid -s UUID -o value /dev/sda3) none luks,discard" > /etc/crypttab
```

20



```

Activities Terminal ▾ Jul 30 14:04
root@lambda-recovery: /
>_ ▾ Q E _ X
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.
# For full documentation of the options in this file, see:
#   info -f grub -n 'Simple configuration'

GRUB_DEFAULT=0
#GRUB_TIMEOUT_STYLE=hidden
GRUB_TIMEOUT=500
GRUB_DISTRIBUTOR='lsb_release -i -s 2> /dev/null || echo Debian'
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX=""

# Uncomment to enable BadRAM filtering, modify to suit your needs
# This works with Linux (no patch required) and with any kernel that obtains
# the memory map information from GRUB (GNU Mach, kernel of FreeBSD ...)
#GRUB_BADRAM="0x01234567,0xfefefefe,0x89abcdef,0xefefefef"

# Uncomment to disable graphical terminal (grub-pc only)
#GRUB_TERMINAL=console

# The resolution used on graphical terminal
# note that you can use only modes which your graphic card supports via VBE
# you can see them in real GRUB with the command 'vbeinfo'
#GRUB_GFXMODE=640x480

# Uncomment if you don't want GRUB to pass "root=UUID=xxx" parameter to Linux
#GRUB_DISABLE_LINUX_UUID=true

# Uncomment to disable generation of recovery mode menu entries
#GRUB_DISABLE_RECOVERY="true"

# Uncomment to get a beep at grub start
#GRUB_INIT_TUNE="480 440 1"
^[

```

edit /etc/default/grub to show the grub menu and rebuild grub

I neglected to update-initramfs before I rebooted

DO NOT REBOOT until you have run `update-initramfs -u -k all` [shown on slide 24](#)

21

```
[ 0.786761] pcieport 0000:20:01.1: AER: aer_status: 0x00000040, aer_mask: 0x0
[ 0.786769] pcieport 0000:20:01.1: AER: aer_layer=Data Link Layer, aer_agent=
Receiver ID
[ 2.424747]
Volume group "2004-vg" not found
Cannot process volume group 2004-vg
/dev/sdb: open failed: No medium found
/dev/sdc: open failed: No medium found
Volume group "2004-vg" not found
Cannot process volume group 2004-vg
Gave up waiting for root file system device. Common problems:
- Boot args (cat /proc/cmdline)
- Check rootdelay= (did the system wait long enough?)
- Missing modules (cat /proc/modules; ls /dev)
ALERT! /dev/mapper/2004--vg-root does not exist. Dropping to a shell!
```

```
BusyBox v1.30.1 (Ubuntu 1:1.30.1-4ubuntu6.4) built-in shell (ash)
Enter 'help' for a list of built-in commands.
```

```
(initramfs) _
```

I included these and the next several slides to show you how to back out of the mistake I made, I thought it might

be helpful

You can see here that /dev/mapper doesn't have my LVM volume group because GRUB did not prompt to decrypt the partition

22

```
Usage: modinfo [-adipno] [-F keyword] MODULE
  -a           Shortcut for '-F author'
  -d           Shortcut for '-F description'
  -l           Shortcut for '-F license'
  -p           Shortcut for '-F parm'
  -F keyword   Keyword to look for
  -o           Separate output with NULS
(initramfs) modinfo luks2
modinfo: can't open '/5.15.0-41-generic/': No such file or directory
(initramfs) modinfo luks
modinfo: can't open '/5.15.0-41-generic/': No such file or directory
(initramfs) modprobe luks2
(initramfs) exit
Give up waiting for root file system device. Common problems:
- Boot args (cat /proc/cmdline)
  - Check rootdelay= (did the system wait long enough?)
- Missing modules (cat /proc/modules; ls /dev)
ALERT! /dev/mapper/2004--vg-root does not exist. Dropping to a shell!

BusyBox v1.30.1 (Ubuntu 1:1.30.1-4ubuntu6.4) built-in shell (ash)
Enter 'help' for a list of built-in commands.

(initramfs) continue
(initramfs) ls
dev      bin      init      lib64      sbin      var       tmp
root    conf      lib      libx32     scripts    sys      cryptroot
kernel  etc      lib32     run       usr       proc
(initramfs) df .
Filesystem      1024-blocks  Used  Available Use% Mounted on
none            0          0        0    0% /
(initramfs) cd cryptroot/
(initramfs) ls
crypttab
(initramfs) cat crypttab
(initramfs) normal
sh: normal: not found
(initramfs) crypt
cryptroot-unlock cryptsetup
(initramfs) crypt
cryptroot-unlock cryptsetup
(initramfs) cryptsetup open /dev/sda3 sda3_crypt
WARNING: Locking directory /run/cryptsetup is missing!
Enter passphrase for /dev/sda3: _
```

From the initramfs busybox environment

```
cryptsetup open /dev/sda3 sda3_crypt
```

this will map the the expected mapper location and boot can resume

23

```

-l           Shortcut for '-F license'
-p           Shortcut for '-F parm'
-F keyword   Keyword to look for
-o           Separate output with NULs
(initramfs) modinfo luks2
modinfo: can't open '/5.15.0-41-generic': No such file or directory
(initramfs) modinfo luks
modinfo: can't open '/5.15.0-41-generic': No such file or directory
(initramfs) modprobe luks2
(initramfs) exit
Gave up waiting for root file system device. Common problems:
- Boot args (cat /proc/cmdline)
  - Check rootdelay= (did the system wait long enough?)
- Missing modules (cat /proc/modules; ls /dev)
ALERT! /dev/mapper/2004--vg-root does not exist. Dropping to a shell!

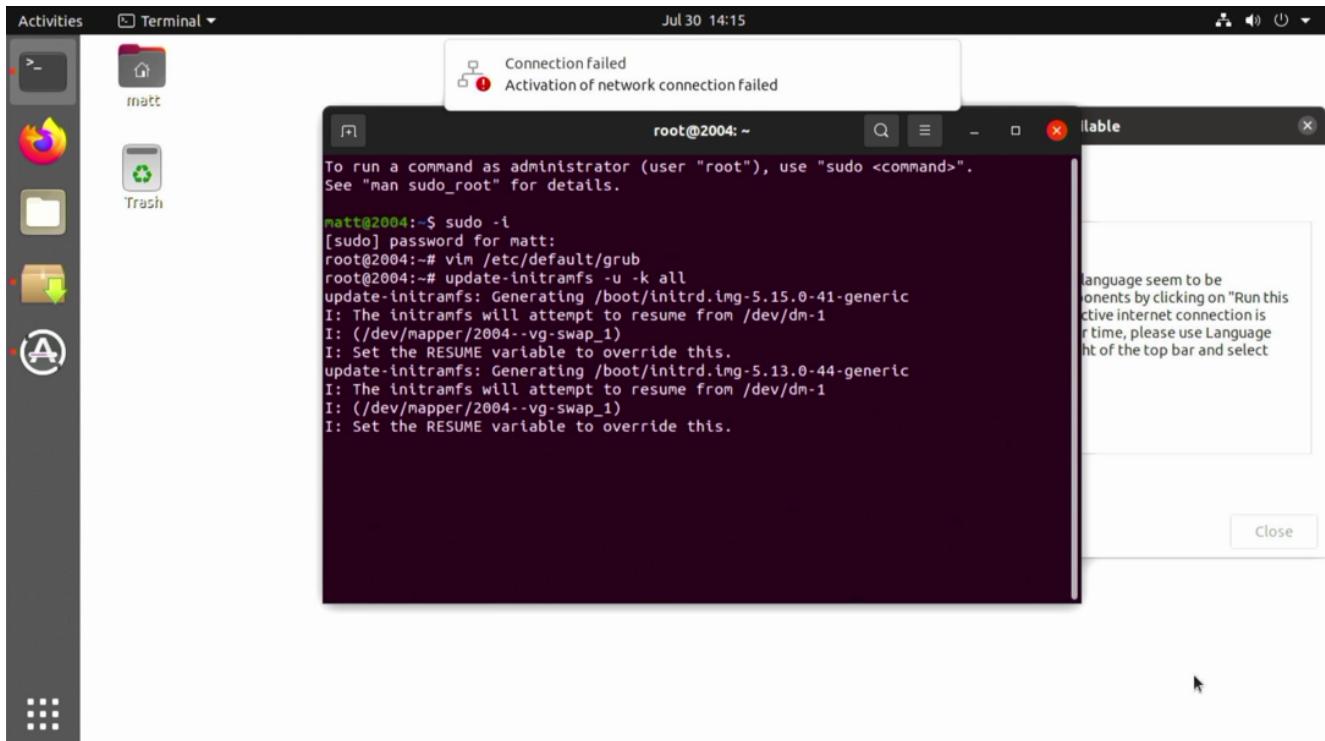
BusyBox v1.30.1 (Ubuntu 1:1.30.1-4ubuntu6.4) built-in shell (ash)
Enter 'help' for a list of built-in commands.

(initramfs) continue
(initramfs) ls
dev      bin      init      lib64      sbin      var      tmp
root    conf      lib       libx32     scripts    sys      cryptroot
kernel  etc      lib32     run       usr       proc
(initramfs) df .
Filesystem      1024-blocks  Used Available Use% Mounted on
none            0          0        0  0% /
(initramfs) cd cryptroot/
(initramfs) ls
crypttab
(initramfs) cat crypttab
(initramfs) normal
sh: normal: not found
(initramfs) crypt
cryptroot-unlock cryptsetup
(initramfs) crypt
cryptroot-unlock cryptsetup
(initramfs) cryptsetup open /dev/sda3 sda3_crypt
WARNING: Locking directory /run/cryptsetup is missing!
Enter passphrase for /dev/sda3:
(initramfs) continue
(initramfs) exit
/dev/mapper/2004--vg-root: clean, 277455/11272192 files, 5672301/45076480 blocks

```

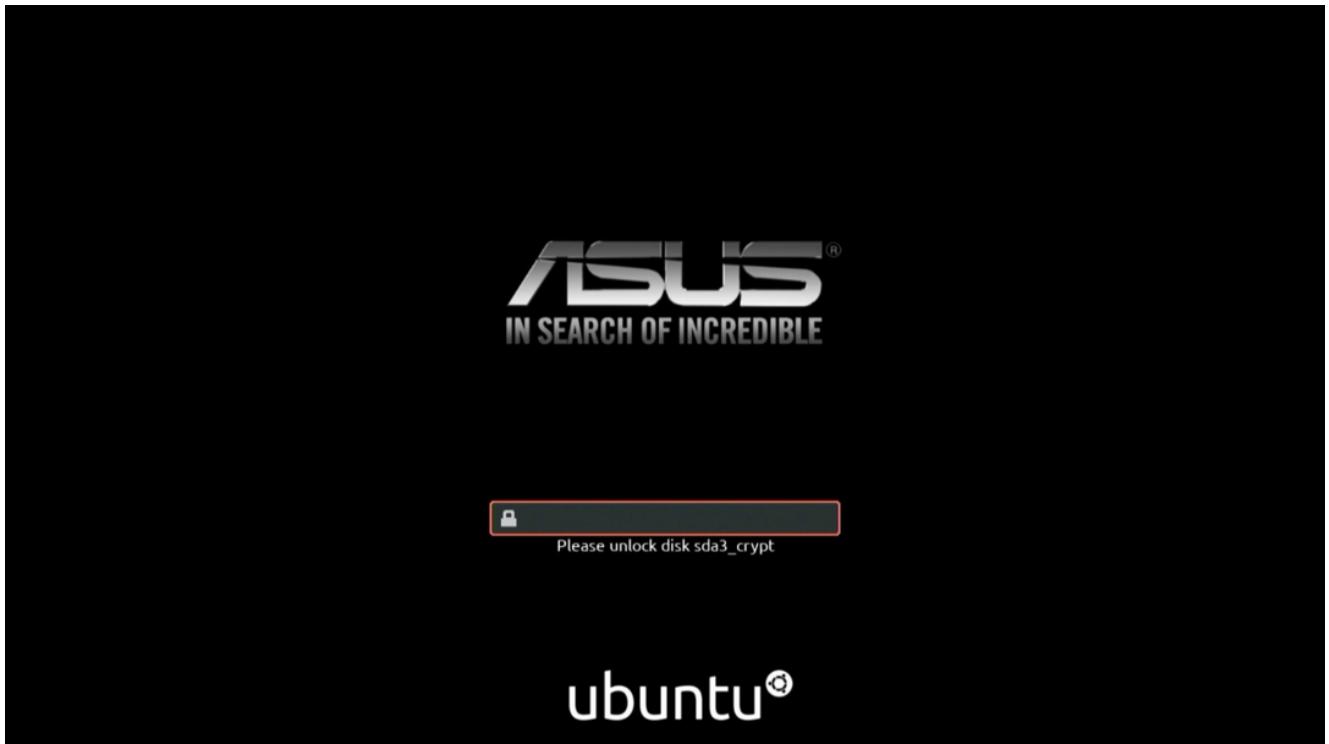
exit to continue boot

24



This is what I should have done on slide 20

25



Reboot and you should be prompted to unlock sda3_crypt

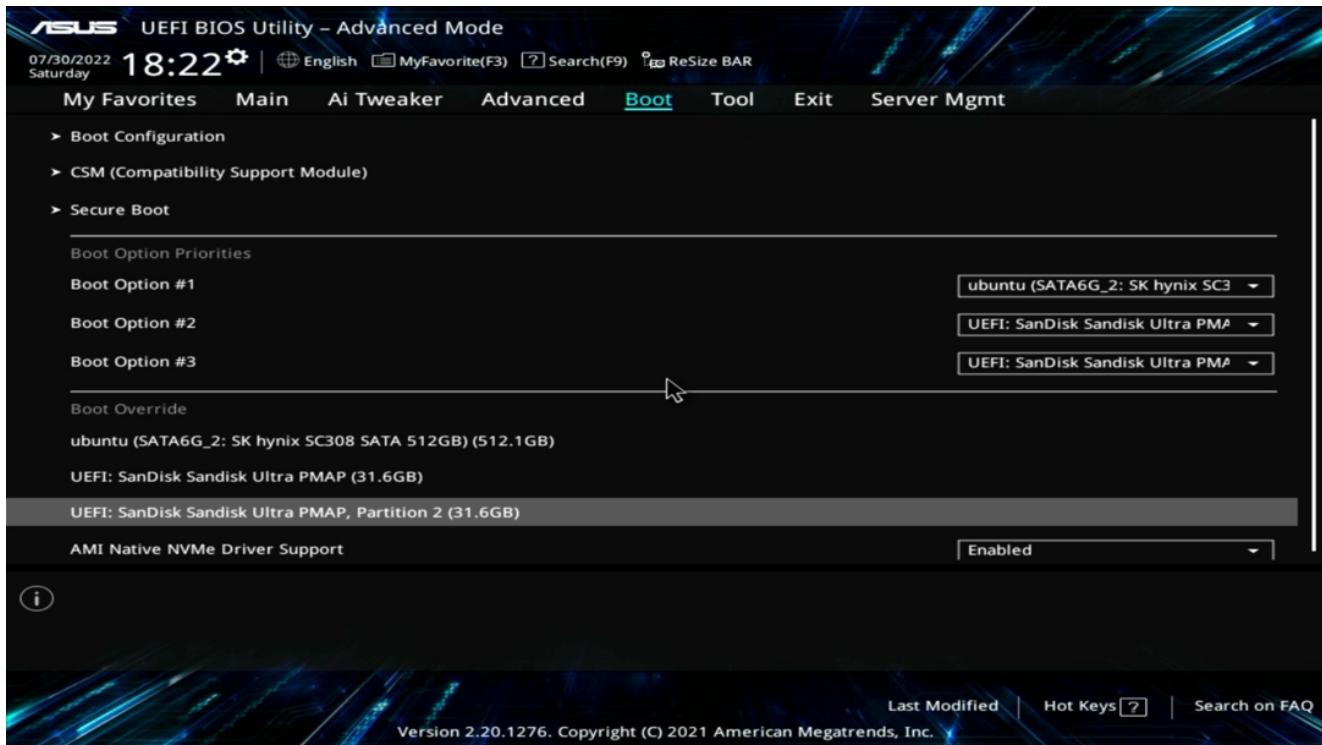
26



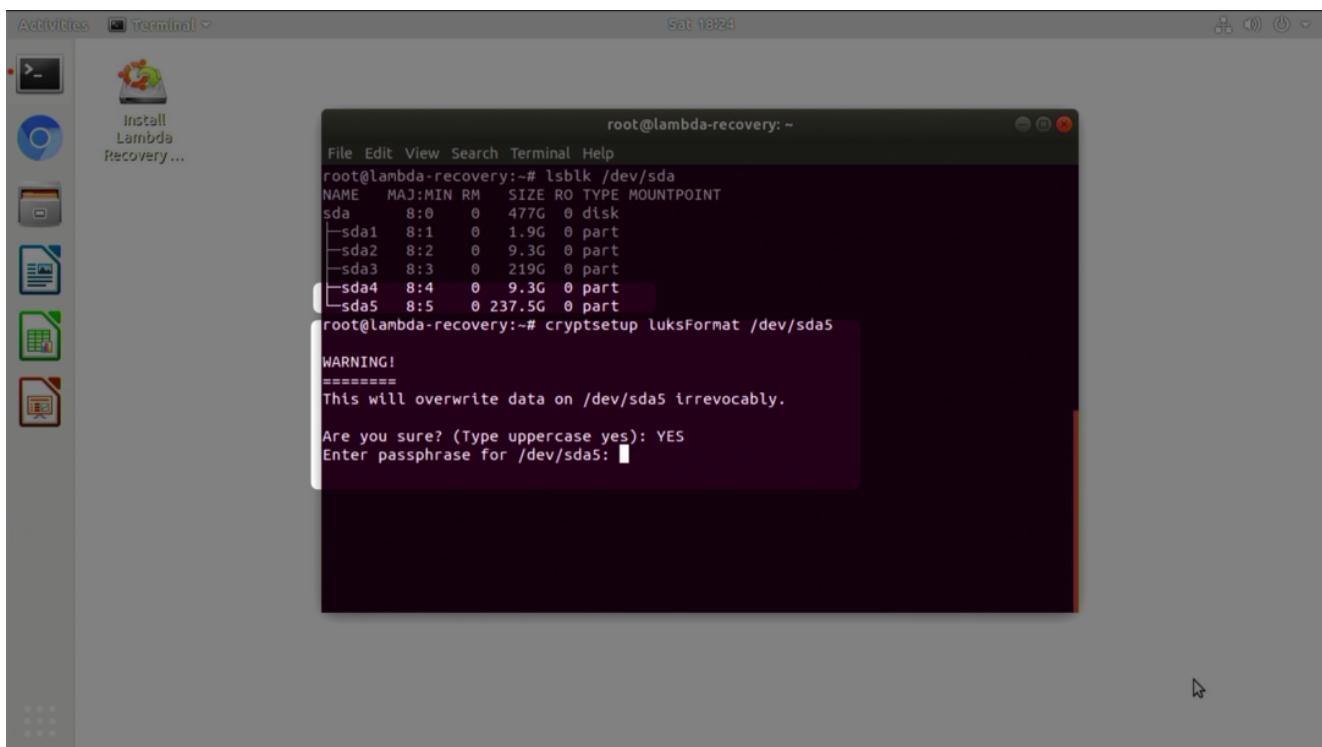
it works

27 reboot into the UEFI/BIOS and boot from the 18.04 Rescue

ISO



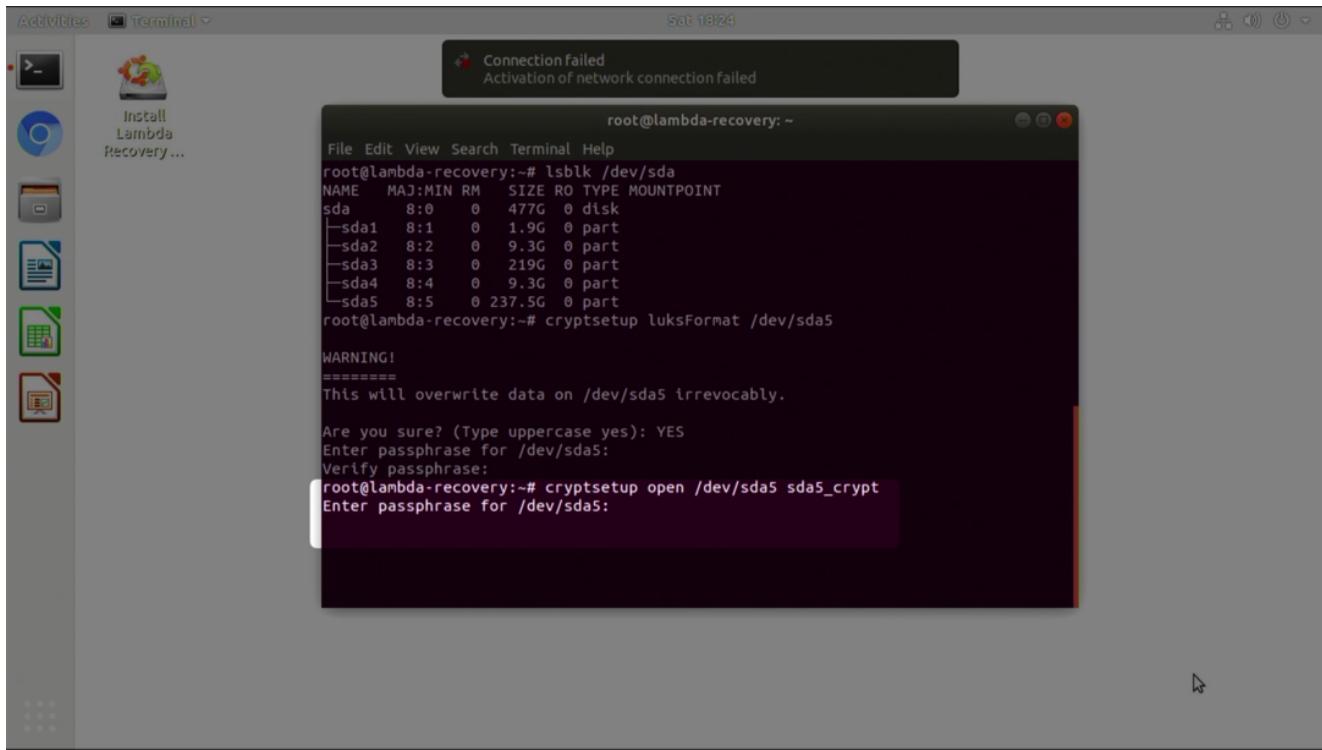
28



similar to what we did for 20.04, but this time we will be using /dev/sda4 for /boot and /dev/sda5 for the LUKS partition

```
cryptsetup luksFormat /dev/sda5
```

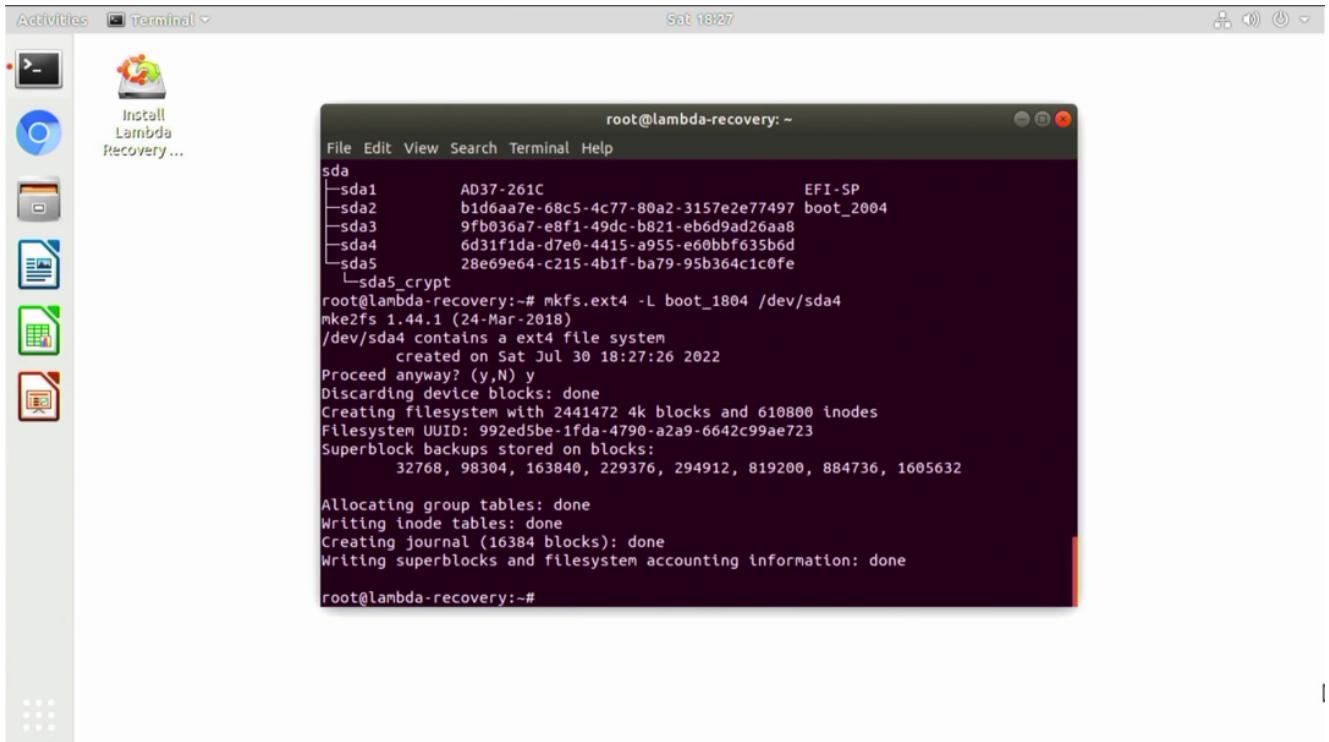
29



open and setup the device mapper to sda5_crypt

```
cryptsetup open /dev/sda5 sda5_crypt
```

30



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window title is "root@lambda-recovery: ~". The terminal content shows the creation of an ext4 filesystem on /dev/sda4:

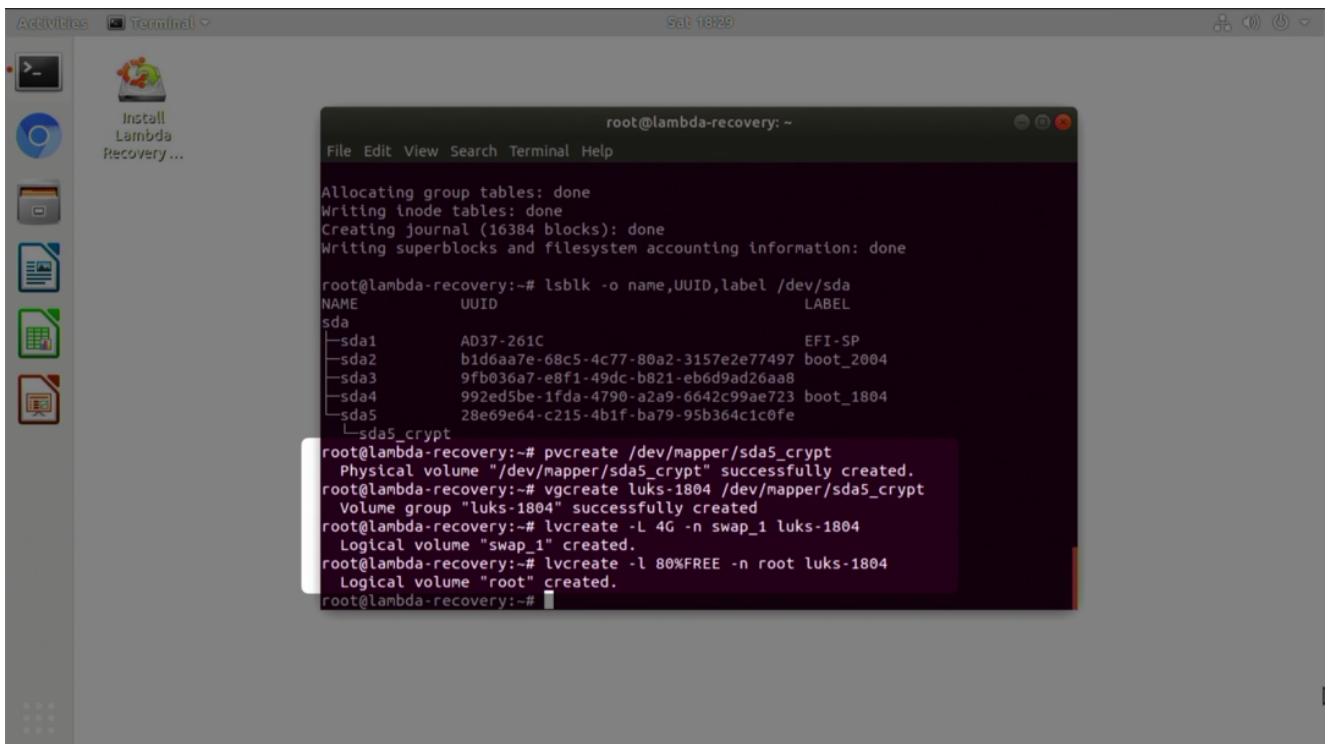
```
sda
└─sda1      AD37-261C          EFI-SP
└─sda2      b1d6aa7e-68c5-4c77-80a2-3157e2e77497 boot_2004
└─sda3      9fb036a7-e8f1-49dc-b821-eb6d9ad26aa8
└─sda4      6d31f1da-d7e0-4415-a955-e60bbf635b6d
└─sda5      28e69e64-c215-4b1f-ba79-95b364c1c0fe
    └─sda5_crypt
root@lambda-recovery:~# mkfs.ext4 -L boot_1804 /dev/sda4
mke2fs 1.44.1 (24-Mar-2018)
/dev/sda4 contains a ext4 file system
        created on Sat Jul 30 18:27:26 2022
Proceed anyway? (y,N) y
Discarding device blocks: done
Creating filesystem with 2441472 4k blocks and 610800 inodes
Filesystem UUID: 992ed5be-1fda-4790-a2a9-6642c99ae723
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
root@lambda-recovery:~#
```

create an ext4 filesystem on /dev/sda4

```
mkfs.ext4 -L boot_1804 /dev/sda4
```

31



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window title is "root@lambda-recovery: ~". The terminal content continues from the previous command, showing the creation of LVM volumes and logical volumes:

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

root@lambda-recovery:~# lsblk -o name,UUID,label /dev/sda
NAME      UUID                LABEL
sda
└─sda1      AD37-261C          EFI-SP
└─sda2      b1d6aa7e-68c5-4c77-80a2-3157e2e77497 boot_2004
└─sda3      9fb036a7-e8f1-49dc-b821-eb6d9ad26aa8
└─sda4      992ed5be-1fda-4790-a2a9-6642c99ae723 boot_1804
└─sda5      28e69e64-c215-4b1f-ba79-95b364c1c0fe
    └─sda5_crypt
root@lambda-recovery:~# pvcreate /dev/mapper/sda5_crypt
  Physical volume "/dev/mapper/sda5_crypt" successfully created.
root@lambda-recovery:~# vgcreate luks-1804 /dev/mapper/sda5_crypt
  Volume group "luks-1804" successfully created
root@lambda-recovery:~# lvcreate -L 4G -n swap_1 luks-1804
  Logical volume "swap_1" created.
root@lambda-recovery:~# lvcreate -l 80%FREE -n root luks-1804
  Logical volume "root" created.
root@lambda-recovery:~#
```

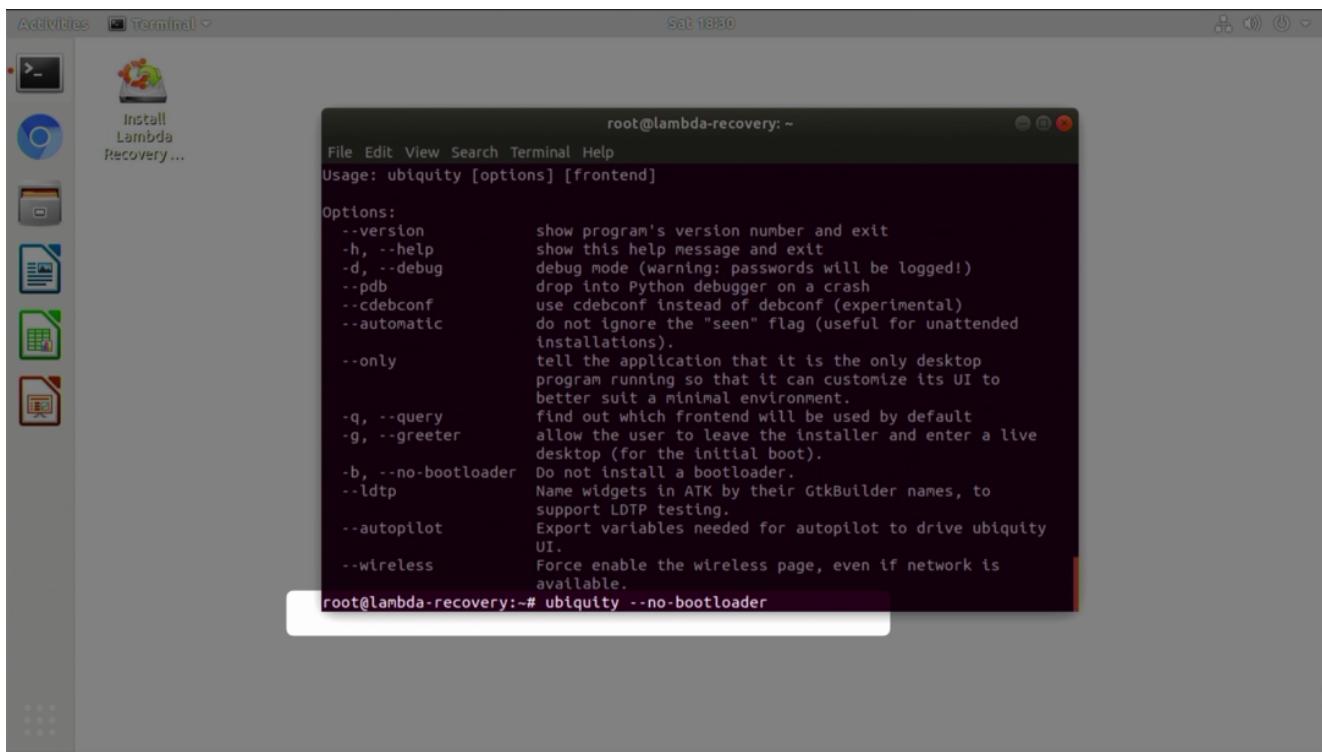
NOTE-- I messed up the LVM naming convention here. What I did works, but 100% of the documentation uses <--vg like i did for 20.04. You may want to sub 1804-vg for luks-1804 in the following commands.

```
pvcreate /dev/mapper/sda5_crypt  
vgcreate luks-1804 /dev/mapper/sda5_crypt  
lvcreate -L 4G -n swap_1 luks-1804  
lvcreate -l 80%FREE -n root luks-1804
```

CORRECTED

```
vgcreate 1804-vg /dev/mapper/sda5_crypt  
lvcreate -L 4G -n swap_1 1804-vg  
lvcreate -l 80%FREE -n root 1804-vg
```

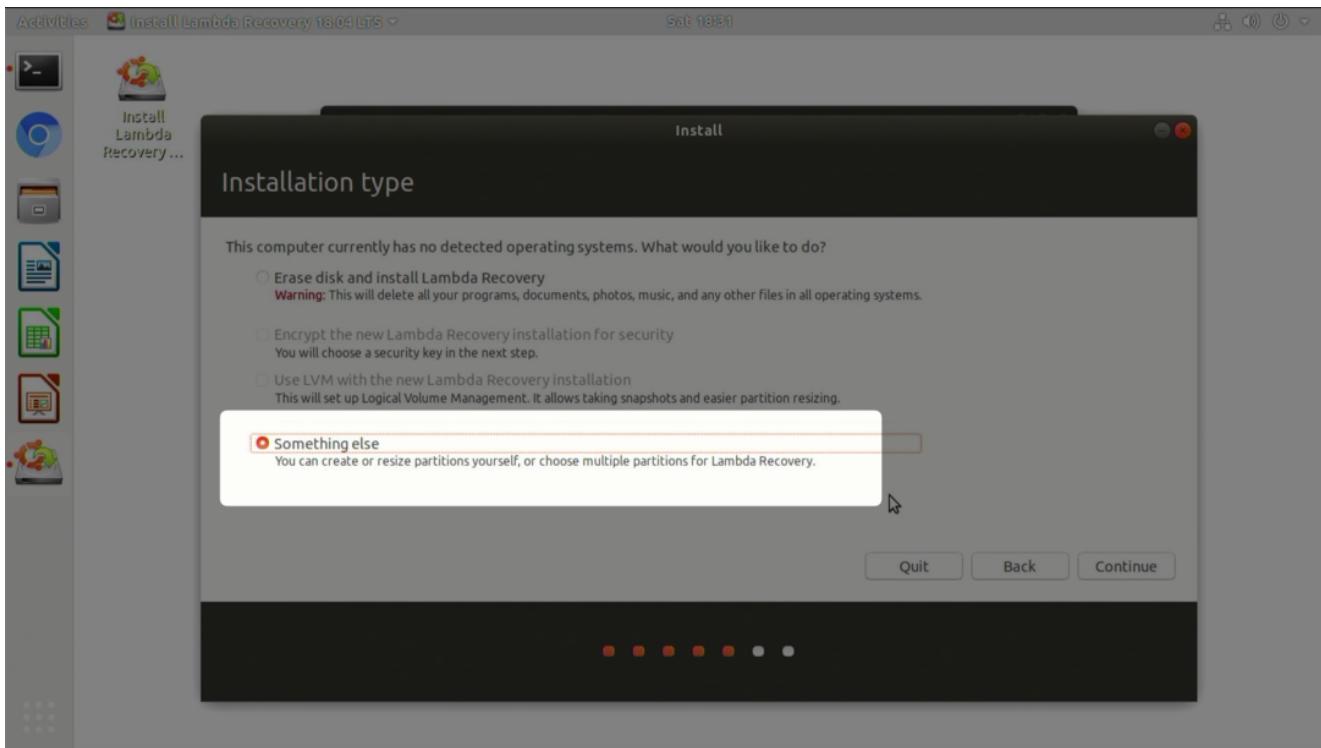
32



launch ubiquity without the installing a bootloader

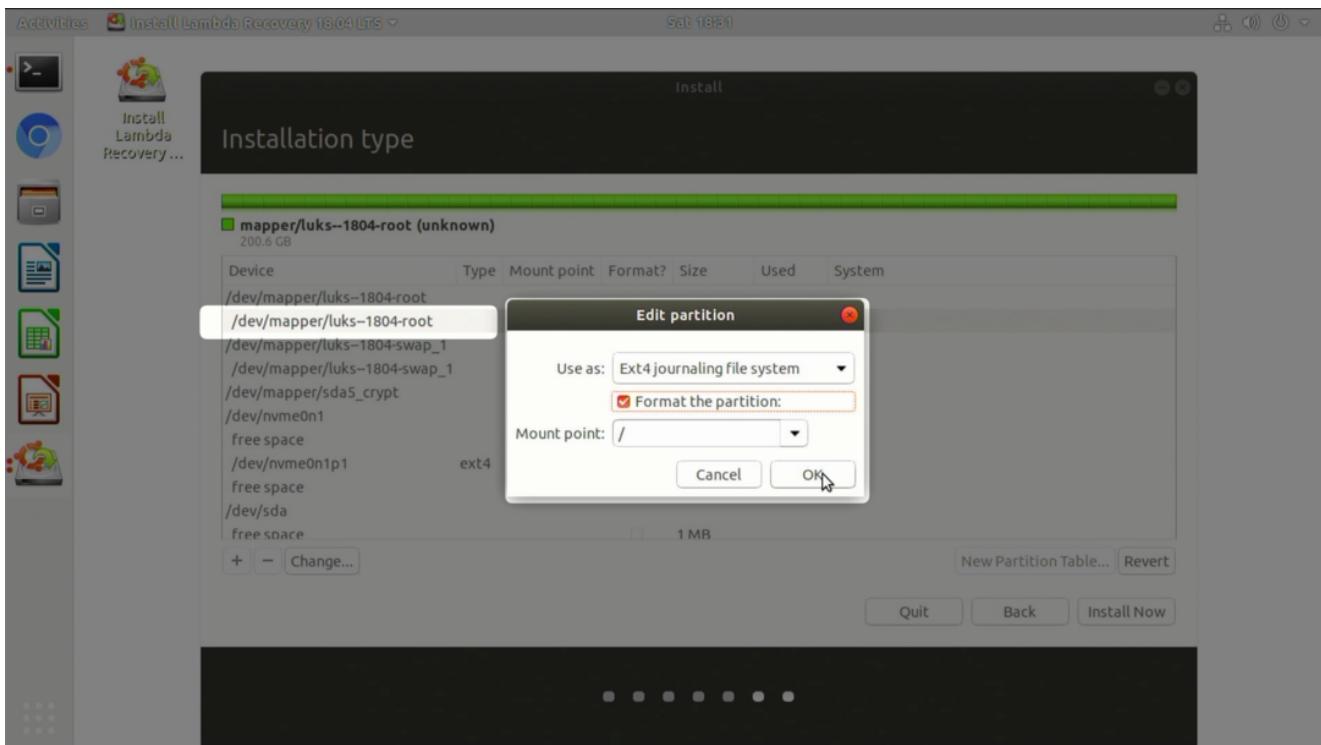
```
uquiquity -b
```

33



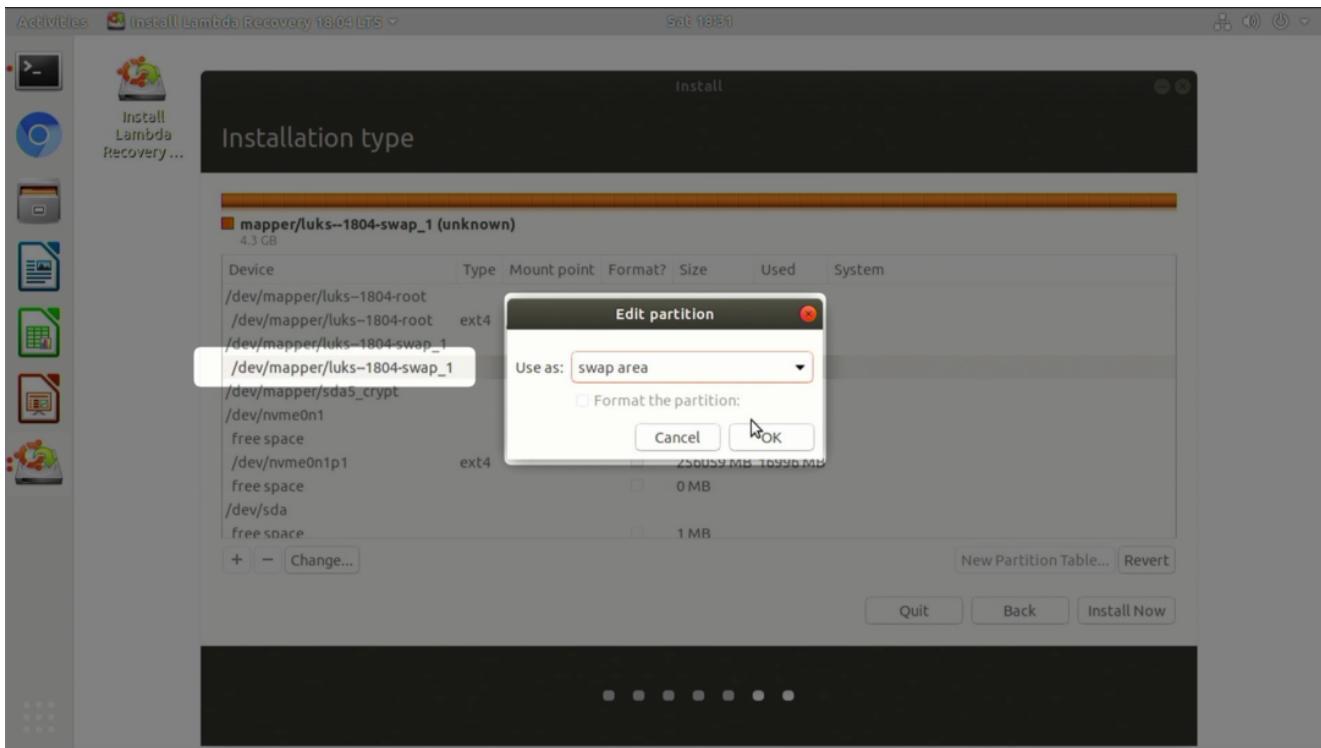
Installation type: Something else

34



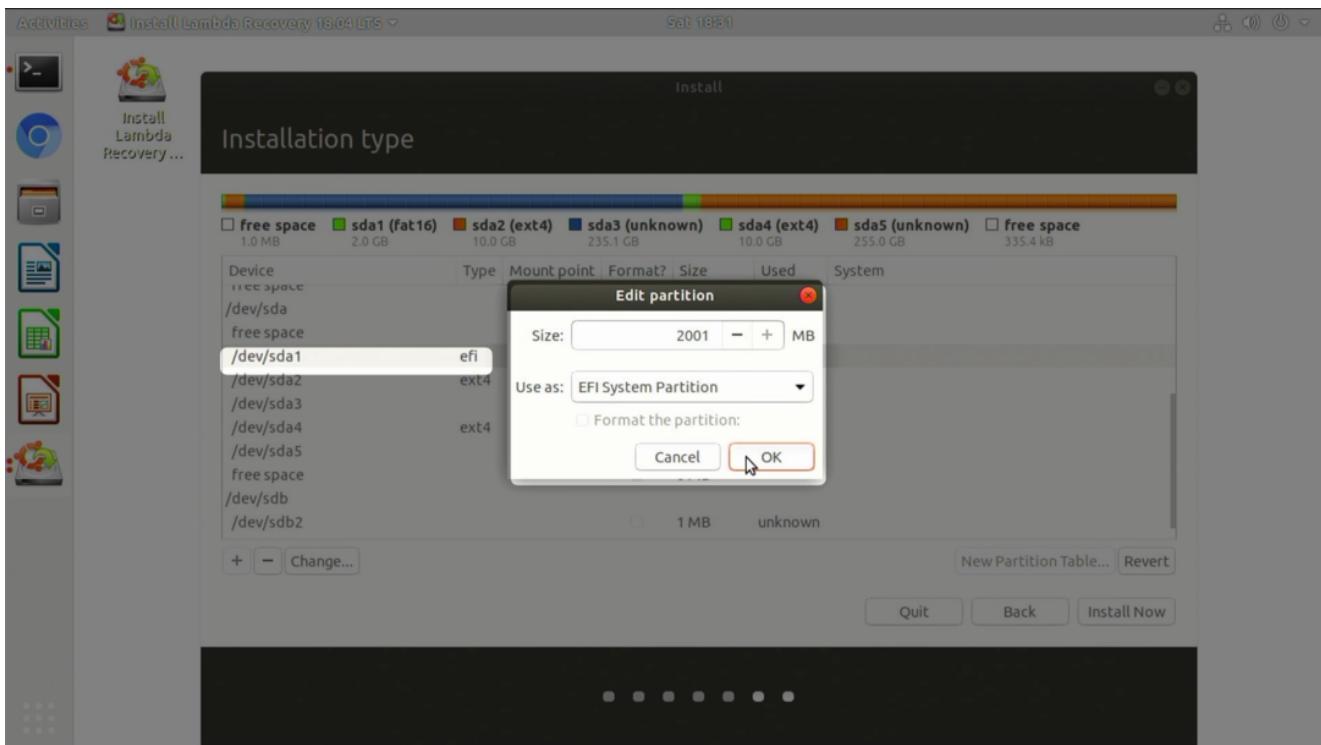
/dev/mapper/luks--1804-root as EXT4, format, Mount Point "/"

35



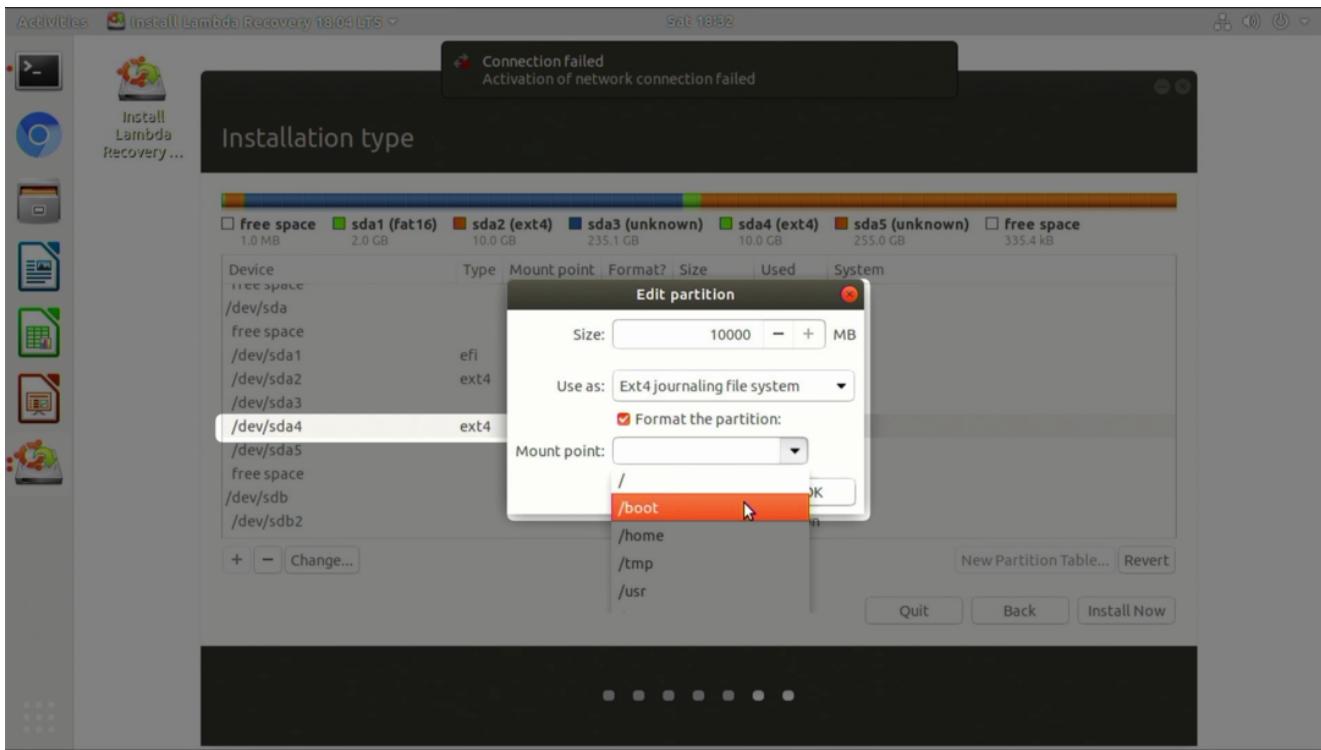
/dev/mapper/luks--1804-swap_1 as 'swap area'

36



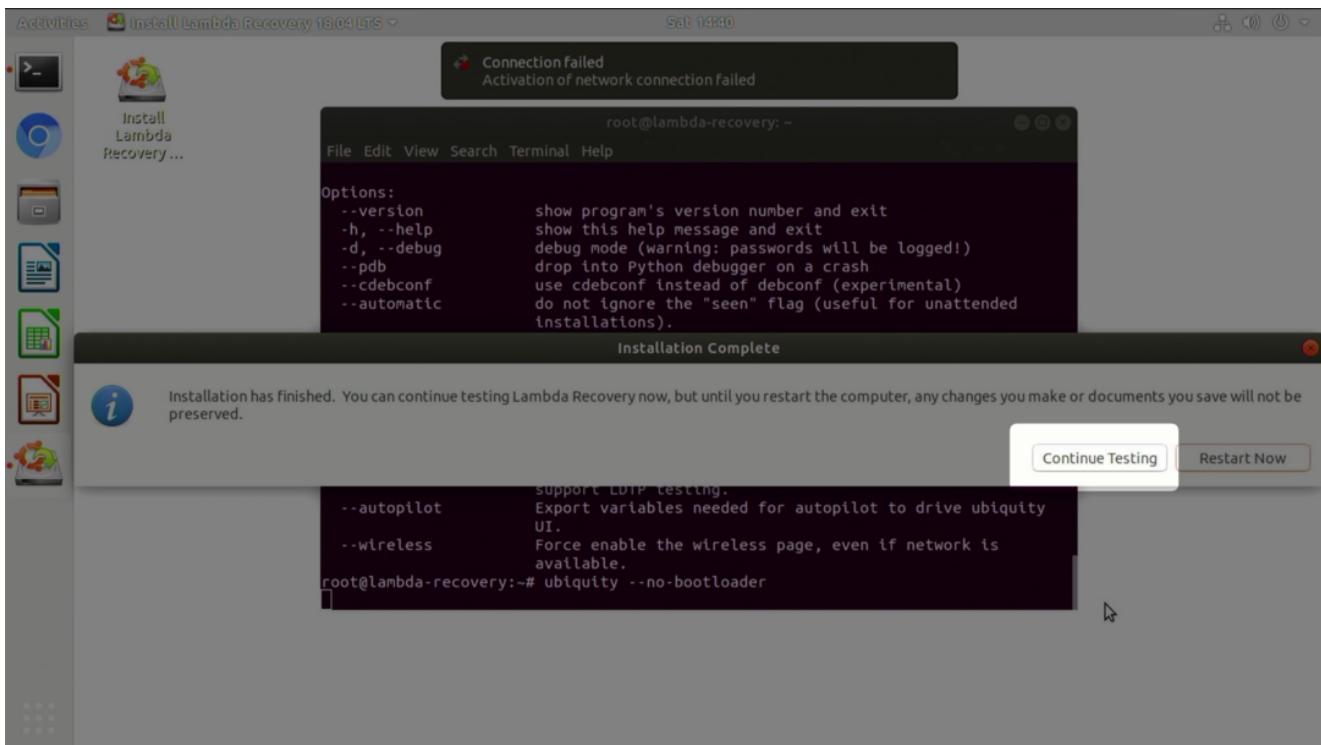
Check that /dev/sda is used as EFI System Partition. No change should be needed here

37



Edit /dev/sda for EXT4, format and mount to '/boot'

38



Install and continue testing upon completion

39

```
root@lambda-recovery: /  
File Edit View Search Terminal Help  
mount: /target/dev: mount point does not exist.  
mount: /target/etc/resolv.conf: mount point does not exist.  
root@lambda-recovery:~# mount /dev/mapper/  
control luks--1804-root luks--1804-swap_1 sda5_crypt  
root@lambda-recovery:~# mount /dev/mapper/luks--1804-root  
mount: /dev/mapper/luks--1804-root: can't find in /etc/fstab.  
1 root@lambda-recovery:~# mount /dev/mapper/luks--1804-root /target  
2 root@lambda-recovery:~# for n in proc sys dev etc/resolv.conf; do mount --rbind  
/S$ /target/$n; done  
3 root@lambda-recovery:~# chroot /target  
4 root@lambda-recovery:~# mount -av  
/ : ignored  
/boot : successfully mounted  
/boot/efi : successfully mounted  
none : ignored  
5 root@lambda-recovery:~# echo "sda5_crypt UUID=$(blkid -s UUID -o value /dev/sda5  
) none luks,discard" > /etc/crypttab  
root@lambda-recovery:~# cat /etc/crypttab  
sda5_crypt UUID=28e69e64-c215-4b1f-ba79-95b364c1c0fe none luks,discard  
root@lambda-recovery:~#
```

18.04 installer does not handle the LVM to /target mount as well as 20.04 so I ran

```
mount /dev/mapper/luks--1804-root /target
```

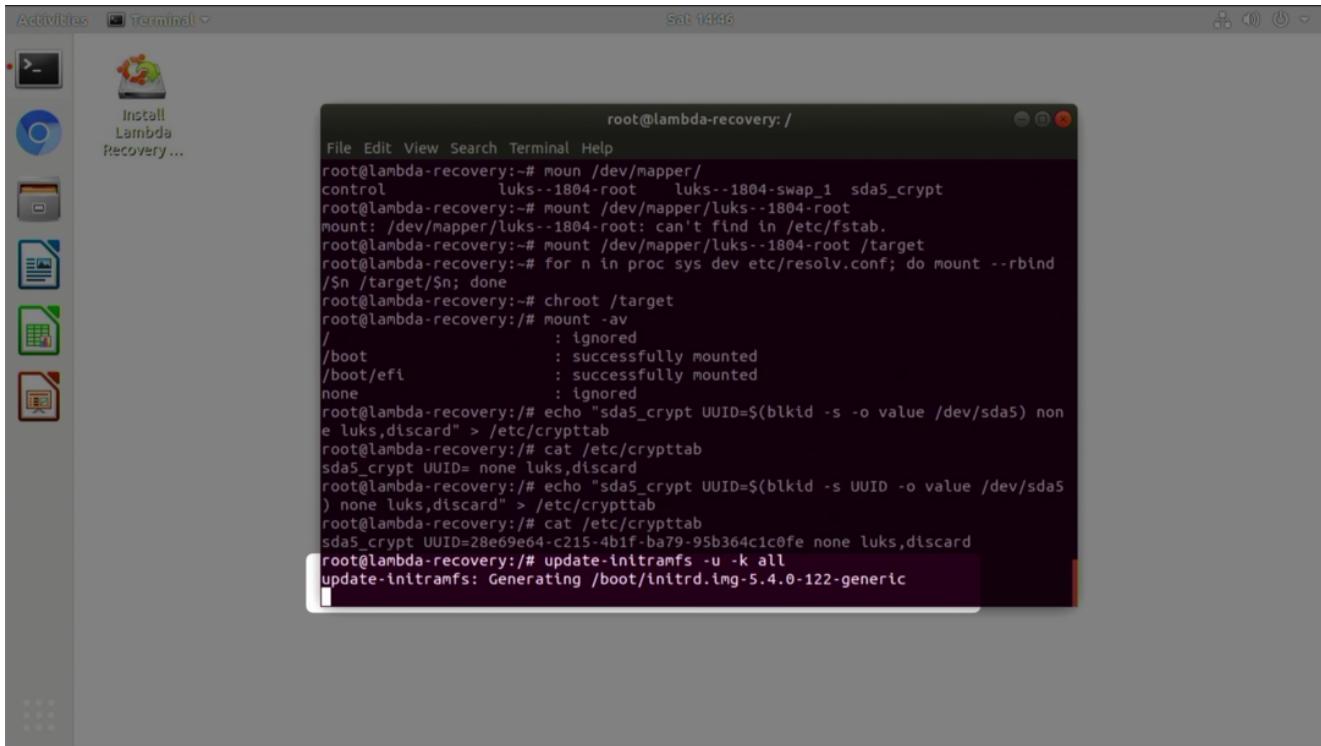
to mount it /target

then:

```
for n in proc sys dev etc/resolv.conf; do mount --rbind /$n /target/$n; done  
chroot /target  
mount -av
```

Then, very similar to for 20.04, just shifting from sda3 to sda5, setup /etc/crypttab

```
echo "sda5_crypt UUID=$(blkid -s UUID -o value /dev/sda5) none luks,discard" > /etc/crypttab
```



A screenshot of a Ubuntu desktop environment. A terminal window is open in the center, showing root access. The terminal window title is "root@lambda-recovery: /". The terminal content shows the process of mounting encrypted partitions and updating the initramfs:

```
root@lambda-recovery:~# mount /dev/mapper/control luks--1804-root luks--1804-swap_1 sda5_crypt
root@lambda-recovery:~# mount /dev/nmapper/luks--1804-root /target
root@lambda-recovery:~# for n in $(grep dev /etc/resolv.conf | awk '{print $2}'); do mount --rbind /$n /target/$n; done
root@lambda-recovery:~# chroot /target
root@lambda-recovery:~# mount -av
/
/boot
/boot/efi
none
root@lambda-recovery:~# echo "sda5_crypt UUID=$(blkid -s -o value /dev/sda5) none luks,discard" > /etc/crypttab
root@lambda-recovery:~# cat /etc/crypttab
sda5_crypt UUID= none luks,discard
root@lambda-recovery:~# echo "sda5_crypt UUID=$(blkid -s -o value /dev/sda5) none luks,discard" > /etc/crypttab
root@lambda-recovery:~# cat /etc/crypttab
sda5_crypt UUID=28e69e64-c215-4bf1-ba79-95b364c1c0fe none luks,discard
root@lambda-recovery:~# update-initramfs -u -k all
update-initramfs: Generating /boot/initrd.img-5.4.0-122-generic
```

`update-initramfs -u -k all`

so that the initramfs will know about the crypttab

** we are relying on 20.04 GRUB so you do not need to update GRUB **

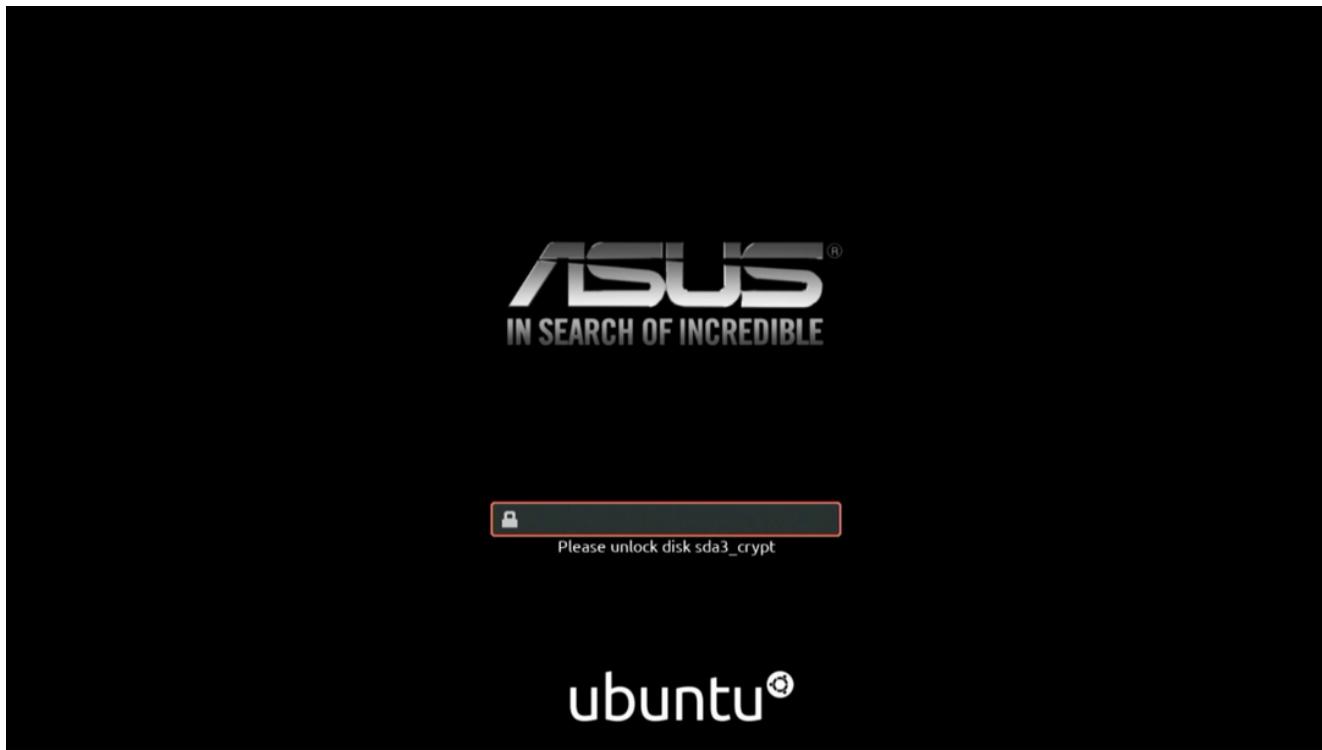
41



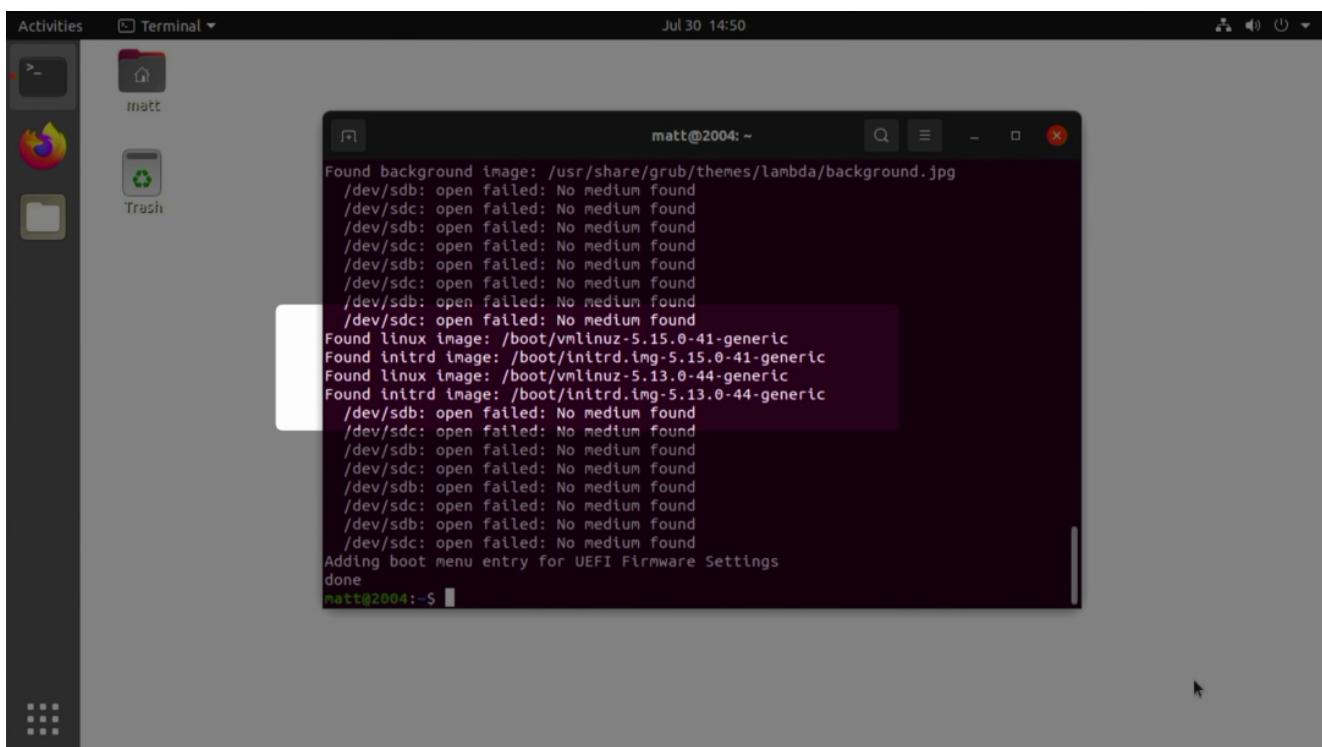
This is the GRUB menu for 20.04-- It doesn't know about 18.04 yet so we have one more cleanup task to perform.

Boot into 20.04

42

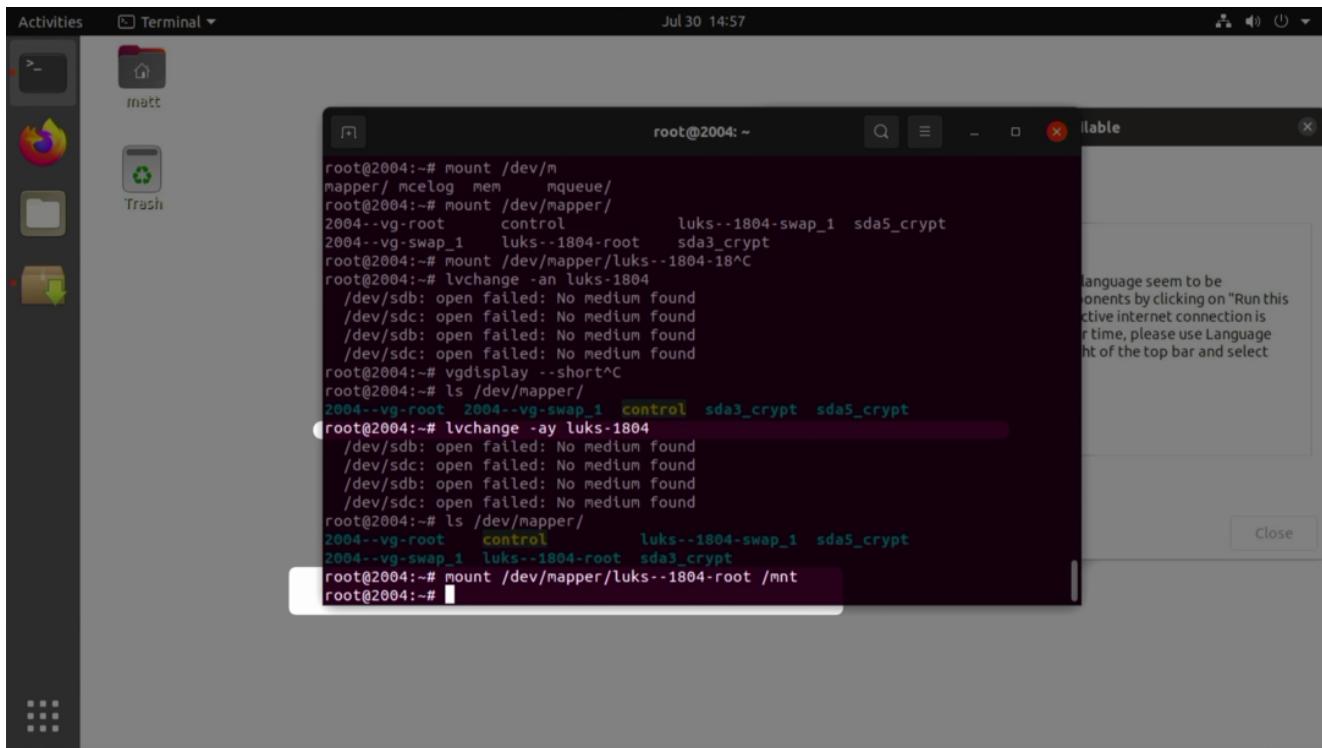


43



update-grub is not finding the initrd images for the 5.4 kernel. Rather than forcing it to, we are going to expose the 18.04 install and run again so it sees it as a separate OS

44



```
root@2004:~# mount /dev/mapper/mcelog mem mqueue/
root@2004:~# mount /dev/mapper/
2004--vg-root control luks--1804-swap_1 sda5_crypt
2004--vg-swap_1 luks--1804-root sda3_crypt
root@2004:~# mount /dev/mapper/luks--1804-18^C
root@2004:~# lvchange -an luks-1804
/dev/sdb: open failed: No medium found
/dev/sdc: open failed: No medium found
/dev/sdb: open failed: No medium found
/dev/sdc: open failed: No medium found
root@2004:~# vgdisplay --short^C
root@2004:~# ls /dev/mapper/
2004--vg-root control luks--1804-swap_1 sda5_crypt
2004--vg-swap_1 luks--1804-root sda3_crypt
root@2004:~# mount /dev/mapper/luks--1804-root /mnt
root@2004:~#
```

** NOT SHOWN ON SLIDE **
decrypt /dev/sda5 and map to expected label of sda5_crypt
cryptsetup open /dev/sda5 sda5_crypt

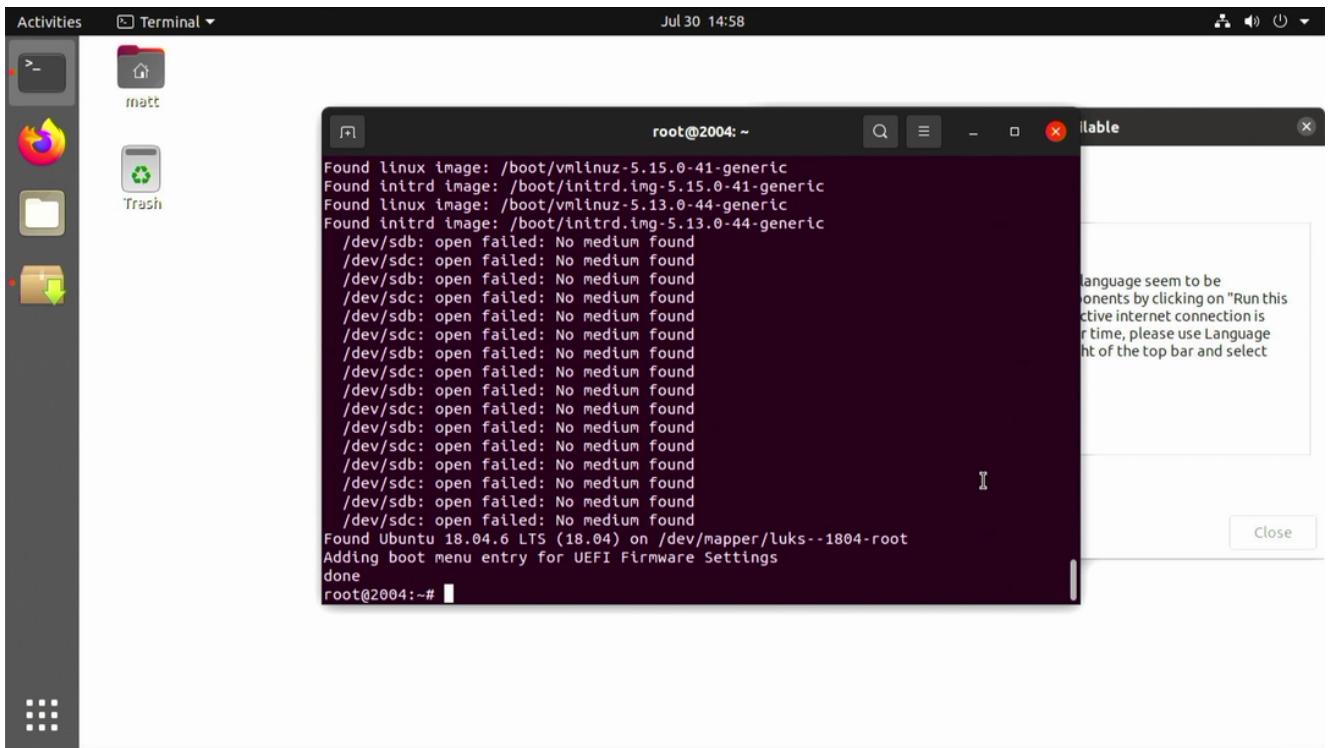
activate root lv

```
lvchange -ay luks-1804
```

mount the lv to /mnt

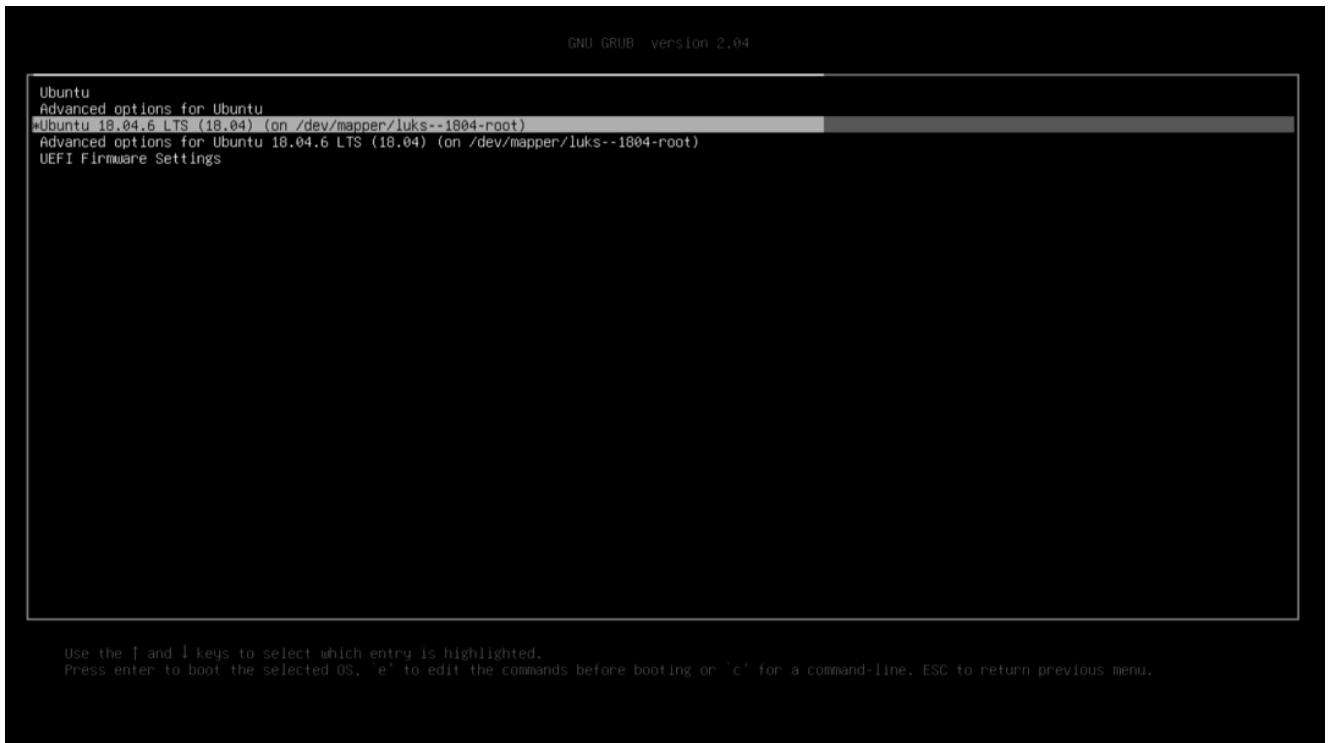
```
mount /dev/mapper/luks--1804-root /mnt
```

45



Run `update_grub` and it will find 18.04.6 and add a boot menu entry.

46



18.04 shows up now

47

```

[ 4.840658] usb 5-5:2: Manufacturer: American Megatrends Inc.
[ 4.841352] usb 5-5:2: SerialNumber: AAAABBBBBCCCCC3
[ 4.932180] usb 5-5:3: new high-speed USB device number 7 using xhci_hcd
[ 5.044599] usb 5-5:3: New USB device found, idVendor=046b, idProduct=fffb0, bcdDevice= 1.00
[ 5.045426] usb 5-5:3: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 5.046200] usb 5-5:3: Product: Virtual Ethernet
[ 5.046893] usb 5-5:3: Manufacturer: American Megatrends Inc.
[ 5.047584] usb 5-5:3: SerialNumber: 1234567890
[ 5.175797] usb 5-5:4: new low-speed USB device number 8 using xhci_hcd
[ 5.299749] usb 5-5:4: New USB device found, idVendor=046b, idProduct=ff10, bcdDevice= 1.00
[ 5.300556] usb 5-5:4: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[ 5.301244] usb 5-5:4: Product: Virtual Keyboard and Mouse
[ 5.301928] usb 5-5:4: Manufacturer: American Megatrends Inc.
[ 5.300022] input: American Megatrends Inc. Virtual Keyboard and Mouse as /devices/pci0000:20/0000:20:01.1/0000:21:00.0/0000:22:08.0/0000:27:00.3/usb5/5-5-
5:4/5-5:4:1.0/0003:046B:FF10.0005/input/input8
[ 5.301615] hid-generic 0003:046B:FF10.0005: input,hidraw4: USB HID v1.10 Keyboard [American Megatrends Inc. Virtual Keyboard and Mouse] on usb-0000:27:00.3-
5:4/input0
[ 5.307951] input: American Megatrends Inc. Virtual Keyboard and Mouse as /devices/pci0000:20/0000:20:01.1/0000:21:00.0/0000:22:08.0/0000:27:00.3/usb5/5-5-
5:4/5-5:4:1.1/0003:046B:FF10.0006/input/input9
[ 5.309687] hid-generic 0003:046B:FF10.0006: input,hidraw5: USB HID v1.10 Mouse [American Megatrends Inc. Virtual Keyboard and Mouse] on usb-0000:27:00.3-5:4/

[ 5.483436] ixgbe 0000:24:00.1: Multiqueue Enabled: Rx Queue count = 32, Tx Queue count = 32 XDP Queue count = 0
[ 5.590721] ixgbe 0000:24:00.1: 31.504 Gb/s available PCIe bandwidth (8.0 GT/s PCIe x4 link)
[ 5.713501] ixgbe 0000:24:00.1: MAC: 4, PHY: 0, PBA No: 000000-000
[ 5.714499] ixgbe 0000:24:00.1: 04:42:1a:f0:26:1e
[ 5.889023] ixgbe 0000:24:00.1: Intel(R) 10 Gigabit Network Connection
[ 5.890831] ixgbe 0000:24:00.1 enp36s0f1: renamed from eth1
[ 5.908717] ixgbe 0000:24:00.0 enp36s0f0: renamed from eth0
[ 5.989779] usb 5-6: New USB device found, idVendor=0b05, idProduct=1984, bcdDevice= 0.21
[ 5.990656] usb 5-6: New USB device strings: Mfr=3, Product=1, SerialNumber=0
[ 5.991369] usb 5-6: Product: USB Audio
[ 5.992077] usb 5-6: Manufacturer: Generic
[ 6.104910] hid-generic 0003:0B05:1984.0007: hiddev2,hidraw6: USB HID v1.11 Device [Generic USB Audio] on usb-0000:27:00.3-6/input7
[ 6.109540] usb-storage 5-5:1:0: USB Mass Storage device detected
[ 6.110369] scsi host6: usb-storage 5-5:1:0
[ 6.111119] usb-storage 5-5:2:0: USB Mass Storage device detected
[ 6.111867] scsi host7: usb-storage 5-5:2:0
[ 6.112577] usbcore: registered new interface driver usb-storage
[ 6.114177] usbcore: registered new interface driver uas
Begin: Loading essential drivers ... done.
Begin: Running /scripts/init-premount ... done.
Begin: Mounting root file system ... Begin: Running /scripts/local-top ...   WARNING: Failed to connect to lvmetad. Falling back to device scanning.
Volume group "luks-1804" not found
Cannot process volume group luks-1804
Please unlock disk sda5_crypt:
```

Prompt to unlock sda5_crypt

48



booted into 18.04

