zedgoat Follow
Documenting my mistakes in the hope I don’t repeat them.
Sep 2, 2017 · 16 min read

A sorta-beginner’s guide to installing Ubuntu Linux on 32-bit UEFI machines.

Update 2018*—since around Fedora 27-ish, it has natively supported 32-bit UEFI for 64 bit installs. This guide still applies for other distributions, but if you want an easier start, try Fedora. See the change note here.*

Update 2018.2—there’s additional workarounds and corrections evolving in the comments, I’ll update the article where I can but take a look before diving in.

. . .

There’s a semi-common edge case that gets around for when installing ubuntu on “BayTrail” hardware, which is found in a lot of low end laptops and tablets. The symptom is, after you’ve created your USB installer, it simply fails to show up in the BIOS—no matter what settings you use.

What’s happening here is:

- BayTrail hardware has a 64-bit CPU, but 32-bit UEFI BIOS, compared to the more common arrangement of both the CPU and BIOS being the same.

- Most common linux distributions now recommend / release as 64-bit.
- Their UEFI BIOS files are also 64-bit.
- The 32-bit BIOS can’t see these 64-bit files.

Hence, you’re stuck.

This is a procedure to help you work around this, and deal with the most common issues you’ll face during the process. There’s plenty of other guides out there, but I’ve found a lot of them to be too technical for beginners, or hard to reproduce in a reliable way.

This is a newbies guide, and kind of verbose.

Some of the terms here aren’t strictly correct, and there may be faster ways of doing some of these steps, but the aim is to walk people through the process in a way that is most likely to succeed, while teaching them a few things along the way. Keep that in mind if any of it offends you as a more expert user!

This could be condensed down to “draw a circle, draw the rest of the damn owl” type steps, but that can cause people to bail out before they have the confidence to try it out themselves.

The prep was done on macOS, but I’ll try and help out for Windows systems.

If I get a hold of a windows machine in future, I'll update it with some extra content, but you'll still be able to use this procedure either way. I'm assuming you're not on Ubuntu already because…beginner's guide.

I'm also assuming you're getting rid of a Windows install from a low end machine, as the process is fairly different for macOS replacements.

The reference machine is a Lenovo 100S (100S-11IBY)

My reference machine here is the Lenovo 100S-11IBY, which has an Atom Z3735F processor and eMMC hard disk. It's only seems to be fully supported as of Ubuntu 17+, earlier attempts failed with trackpad and WiFi issues.

. . .

Prerequisites

- The laptop or workstation you want to install on.
- A spare USB stick to make the installer on.
- About a half a day in time
- An EFI file from hirotakaster on GitHub

A quick read over the following isn't a bad idea either (don't worry too much about the details, we're going to walk through the specifics below):

- Installing Ubuntu on Baytrail Tablets Version 2
- How to repair, restore, reinstall grub 2 with a ubuntu live CD

. . .

Linux choice, and some notes on compatibility.

Just get ubuntu.

I’ve based this guide around Ubuntu 17.10, Artful Aardvark, as pulled from the Ubuntu cdimage server in September 2017.

I recommend ubuntu because if you go googling for help, you’re going to find that it dominates the results, even for more general linux questions. It doesn’t hurt to stay mainstream when you’re starting out, because it’s more likely that i) you’ll find answers to your questions and ii) the more mainstream distributions are going to get hammered by a lot of users, find, and solve problems a lot sooner than smaller ones.

Once you have your confidence up, there’s nothing wrong with branching out, but keep it simple for the start, because…

There will be issues, often fatal.

The diversity of linux distributions and hardware out there is really wild west territory. This guide might not work for you at all. Also, by the end of it, most of your hardware will work—but there will probably be at least a few issues with shortcut keys, sleep, display brightness,

battery indication, and a host of other things common to linux, especially on laptops.

This isn’t to knock it but to set your expectations, so be patient and remember to have fun.

Don’t do this on your main computer!

Really, honestly, if you have a daily requirement for a fully supported and working computer, don’t use it for this—buy a cheap, second hand laptop or use the VM or Cloud approaches mentioned below to test. If you are a beginner with linux on your main machine, you need to be ready for some downtime and evenings spent resolving issues, even if you get it right the first time.

You could also just use a VM, Container or the Cloud!

On that note, if you want to learn about linux, but can’t afford a second machine to test things on—you still have options. You can use a virtual machine, such as Virtualbox, Vmware Player or Veemu for free and run it there.

If your current machine isn’t powerful enough to run a VM, you can also host it in the cloud on AWS, gCloud or Azure VM services. For most cases, this is completely free, and equally as capable to get you setup on a server or workstation for the purpose of learning.

Other than the links above, I’m not going to cover these approaches in detail here, but they’re all good options if you need them.

. . .

A summary of the process

You’re going on a bit of a non-standard journey to a working Ubuntu install at this point, and there will be some parts where things look totally broken, so keep in mind the following sequence as we go:

- We’re going to build a standard USB installer.
- Then add an extra file to help it work with the 32-bit BIOS.
- Setup your computer to let you boot from a USB.
- We’re then going to boot to a “live desktop” and see how things look.
- Then, we’ll do an install **which is going to fail at the last step.**
- After that, we’ll start up the USB installer again with a few tricks to get us into the broken installation.
- Finally, we’ll do a little work to fix things up so you can use your Ubuntu install without any hassle.

So, keep in mind—things will sometimes look very broken during this process, but that’s ok—it’s by design, sorta.

. . .

Building a USB installer.

Most linux-es install using a USB installer, which is made by loading an image file onto a USB stick and then booting from it. This is well

covered, but I'm going to summarise here all the same.

Download the ISO

As I noted above, I've grabbed the latest ISO from the Ubuntu site. I'm always going to pick the current build of what's called a "Daily-Live" image, which has the latest drivers and software. They can be less stable, but it gives you a better chance of having all of your hardware work.

When you look at the files here, you want to get the "xxxx-desktop-amd64.iso", where xxxx will be the current "codename" for Ubuntu, at the time of writing this was "Artful".

Format your USB stick

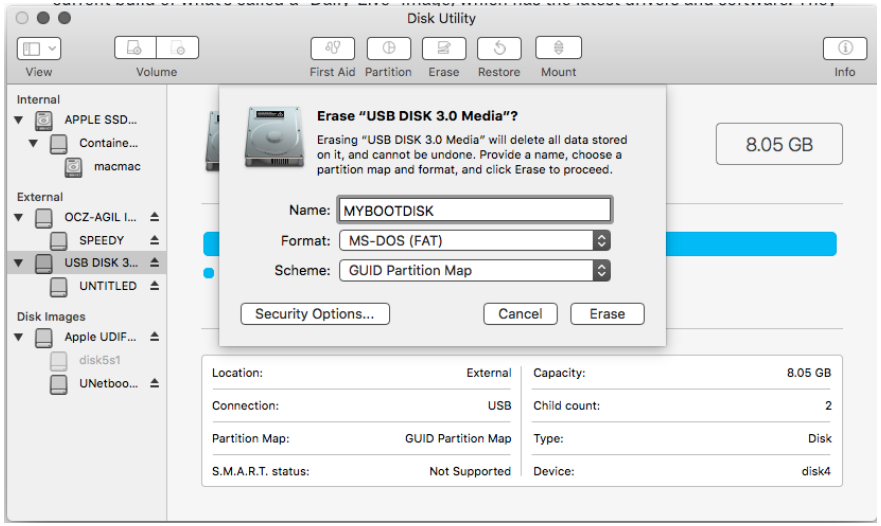
This will change between operating systems, but you want to insert your USB stick and format it as FAT32 using a GUID partition scheme. The short version is, this is what's required for the BIOS to be able to see the disk correctly.

I'm on a mac, so I'm going to detail that process, but link out to some alternatives guides for Windows, and again, assuming you don't need a Ubuntu or Linux how-to at this point.

On Mac

For mac, you can launch the Disk Utility from spotlight, find the disk on the left hand side of the windows, then select "Erase" and choose the following options.

Be careful to find the correct disk! If in doubt, pull the USB drive out of your mac and see which one disappears from the list—that’s the one you want to erase.



On Windows

If you’re using windows, you can just skip ahead to the section on “Creating a bootable USB”, where I’ll provide a guide to using an app called “Rufus”, which is easier than trying to navigate the windows tools for converting to GPT.

If you’re still keen on doing it in windows, you want to start the Disk Management tool by pressing the Windows Key + R, typing diskmgmt.msc and confirming the security prompts as required.

After this, you can follow a guide like this one on Microsoft’s Technet and convert the disk. Again, if this is a bit intimidating, just skip to the

next section, I’ve got you covered.

. . .

Creating a bootable USB

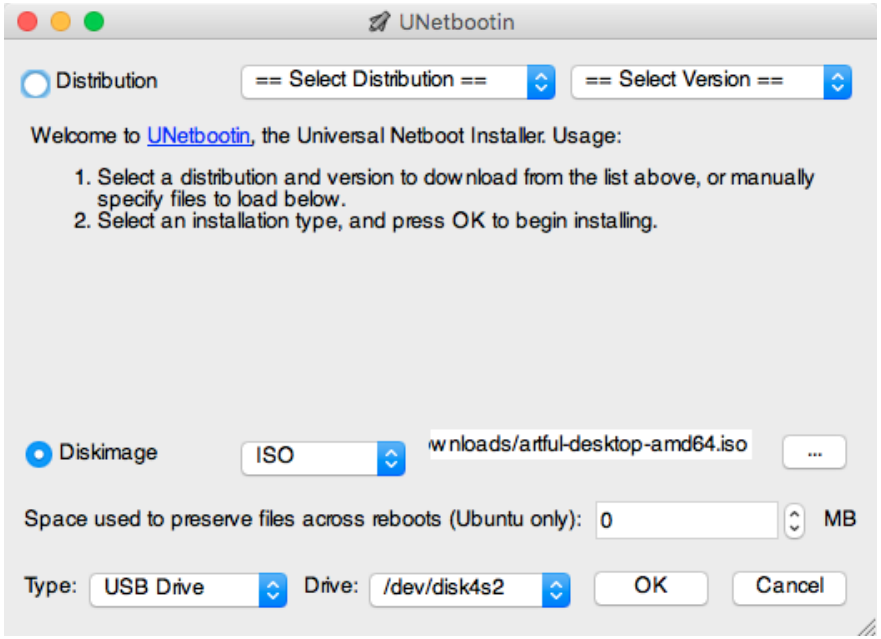
The app choices are specific here as they “unpack” the install ISO onto a disk. There’s other utilities that load the image (such as Etcher, which is my favourite disk utility ever), but you won’t be able to easily change the disk after it’s written, which is important for what we’re doing here.

Remember—the USB stick you use for this will be wiped! Make sure you don’t have anything on there you want to keep.

On Mac with uNetBootin

Grab uNetBootin from github, and add it to your apps. It’s a utility that will take the ISO we downloaded earlier and “extract” it onto the USB stick.

Launch it, and set the options as below. You want to select Diskimage, and then choose the downloaded ISO.



It will have automatically selected a USB disk to write to—careful with this, if you aren't confident confirming which disk it's going to use, you should remove any other USB drives from your computer before doing this step.

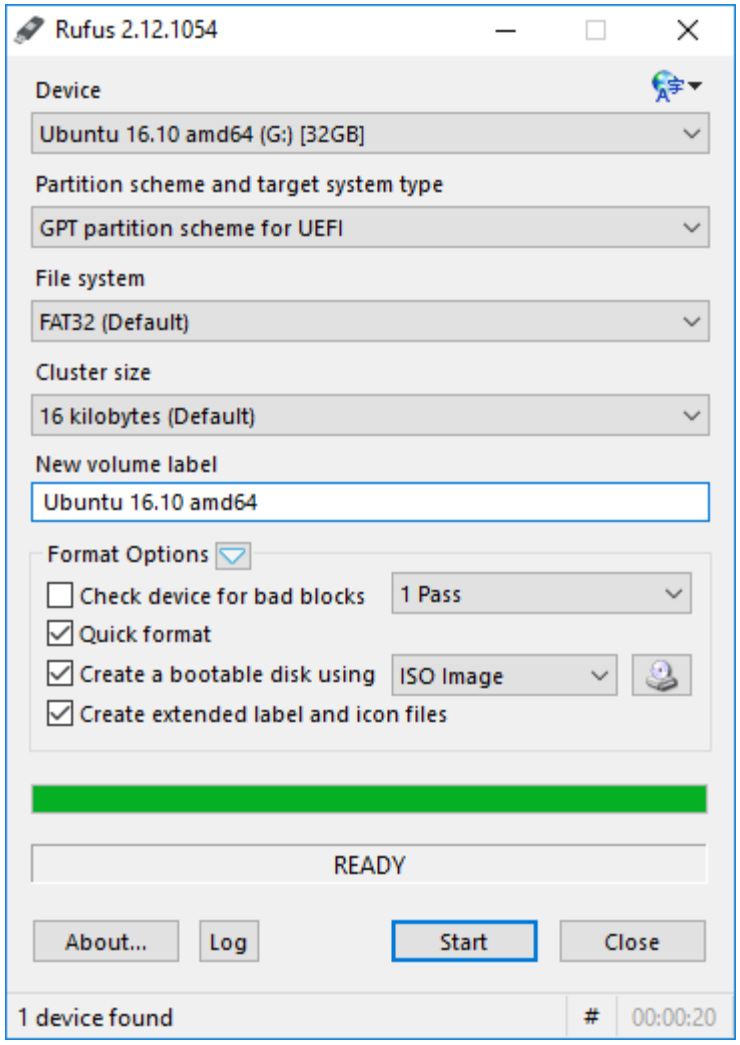
Press OK and let it run until it's completed.

On Windows using Rufus

uNetBootin exists for Windows as well, but I'm recommending Rufus here because it can automatically handle the formatting step for the USB as well, saving you the trouble of doing it manually.

You can grab rufus and install it from here. The screenshot on the website is pretty close to the setup we're going to use to write the USB.

The device section shows the USB stick we’re going to wipe, and the “Create a bootable disk using” section lets you select the downloaded ISO.



. . .

Adding the 32-Bit EFI file.

So, this next bit is important, and relates to the issues outlined at the start of the process. As it turns out, the workaround for the 32 vs 64 issue is pretty straightforward—we can just add the 32-bit EFI file to the Ubuntu installer files and the BIOS will be able to see it.

So, on the disk we've just created there's going to be a /EFI/boot directory. You want to copy the file that we downloaded from github earlier into this directory along side the other files.

After this, the USB is ready to go.

. . .

Setting up your BIOS for the install.

Here be dragons! This is the first step that is potentially destructive to your PC, especially if you're using Windows and Bitlocker. This is also going to be different from machine to machine, so I'm providing generalisations here—you will need to do some googling for your own setup.

A Warning

From here on in, we start making changes to your PC that you might not recover from.

Changing secureboot settings, especially with Windows 10 and Bitlocker (a type of drive encryption), can get you to a point where you're essentially locked out of your operating system.

It’s worth noting down any settings you change here so you can set them back the same way later—there’s still one more step after this where we can quit and stick with Windows, at which point you’ll want to clean up any changes you’ve made.

Having said that, I’m assuming if you go further in this guide that you’re OK with the risk and ready to persist until you have a working setup again. I’ll including a section at the end on how to re-install windows all the same.

The process

Generally, you’re going to do the following:

- Reboot your PC
- Access the BIOS using a shortcut key.
- Ensure UEFI boot is turned on (not legacy).
- Disable secureboot.
- Optionally, reset the BIOS to “install mode” as required.

After this (if everything is setup ok), you will put the USB drive in, restart the machine again, go back into the startup menu, select “Boot Disk” or a similar option, and select the “Ubuntu” entry. This will boot you into the Ubuntu “GRUB” environment.

In the specific case of the Lenovo 100S, this process looks like:

- Turn off the machine

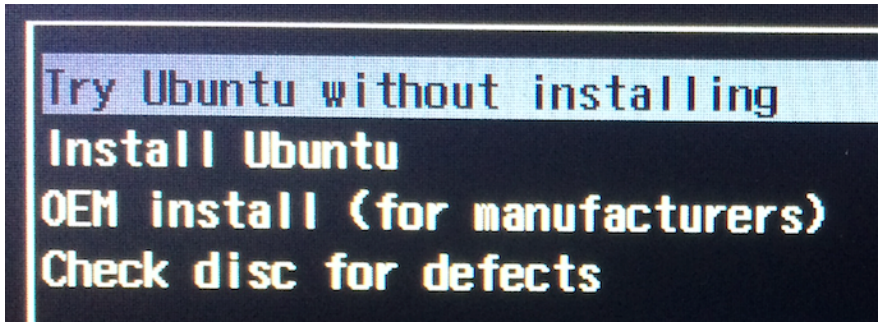
- Press the recovery button next to the HDD light.
- Select BIOS Setup
- In configuration, turn off Secure boot, select Reset to Setup Mode.
- Press F10 to save and exit.
- Turn off the machine again, press the recovery button.
- Select Boot Menu
- Select Ubuntu under EFI Boot Devices.

. . .

Booting to the live desktop.

Ok, so now we’re going to boot to the “Live Desktop” of Ubuntu, which is like a preview of what things will be like once it’s installed, without having changed anything yet.

You should roll through to the first menu automatically. The 32-bit GRUB menu is still text based so you’re going to see the following (apologies for potato quality shot):



We want to Try Ubuntu—we could just go directly to the install, but trying it out gives us a chance to see if everything is working OK, and if we want to go ahead with the install. The try option will take you straight through to a working desktop that is more or less the same as what you’ll have once we finish the install.

If you’re happy with how it looks, you can still trigger the installer from this step.

Testing things out.

Now to test things out. Generally, you want to get a feel for if you’re happy with how everything works. Consider the following:

- Keyboard and Keys.
- Trackpad.
- Wifi and Bluetooth.
- Shortcut keys or other functions.
- Battery life indication.

- Sleep and Wake.
- Etc.

Obviously the applications are going to be different from Windows or macOS, but if there are any major show stoppers (eg, keyboards not working), this is the point where it’s easiest to roll back from.

. . .

Doing the install

A Final Warning—at this step, we really start breaking stuff.

There are way’s to “dual boot” a system between Windows and Ubuntu, but for this guide I want to keep it simple so we’re sticking with a lot of defaults in the installer—one of which involves wiping the existing Windows installation.

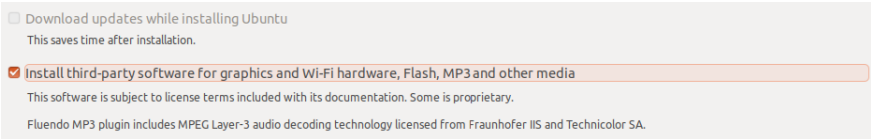
This really is point of no return territory, so make sure you’re happy with how the live desktop is working before going further, and backup anything important on your PC.

Installing Ubuntu

The live distribution has an installer link on the desktop:

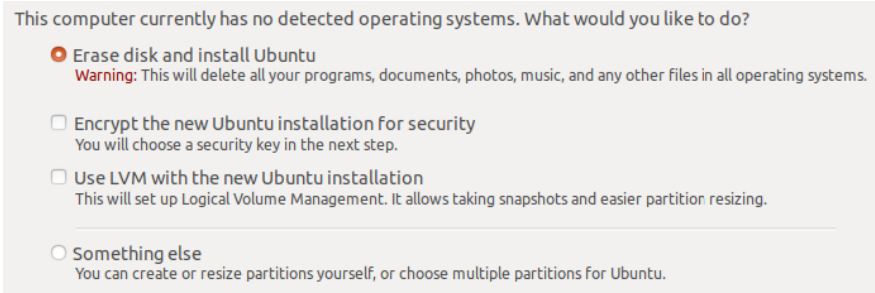
We’re going to start that up, and for the best part, choose the default settings—it’s easier, and customisation tends to put you further away from the mainstream, making support harder.

I’m going to recommend installing third-party software—it can help ensure a smooth experience once Ubuntu is installed. Proprietary drivers are a little bit against the spirit of open source, but in general my preference is towards a better user experience.

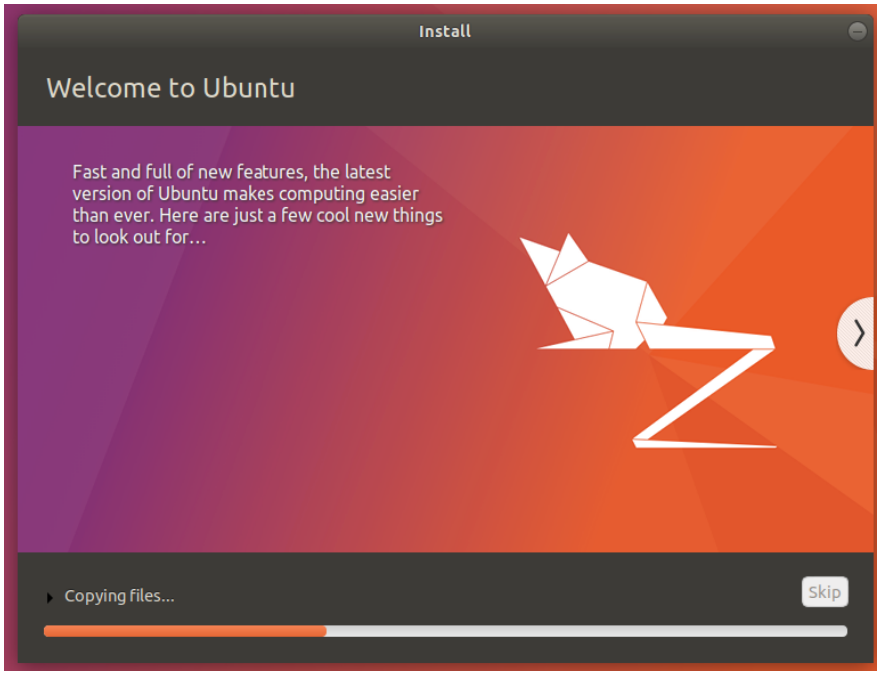


On the next screen, there will be some variations depending on what you already have installed on your PC (I’m taking these screenshots from a VM), regardless, choose the “Erase Disk and Install Ubuntu” option. This automatically chooses a lot of settings that can be hard to get right if you do it yourself.

Choosing the encryption option doesn’t hurt if you want higher security with your install.



Click Install Now when you’re ready to proceed, and the process will start.



This will take about 15–20 mins to complete, and in the meantime you can choose your login details and timezone.

The grub installer failure

Now, the majority of the time I’ve done this, the Grub installer fails, with something along the lines of “grub-efi-ia32 package failed to install to target”.

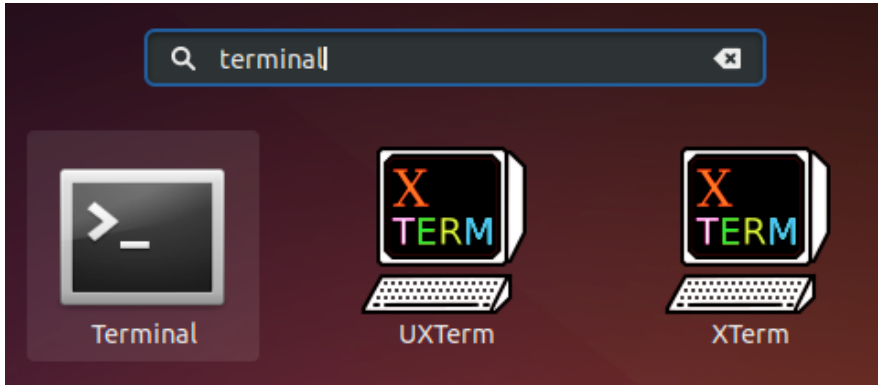
Because we’ve pulled a fast one on the system to get it to work with the 32-bit BIOS, it get’s a little confused at this step. It’s actually ok—we’re going to fix it in the next step with some more trickery.

Confirm the error message and move onto the next step.

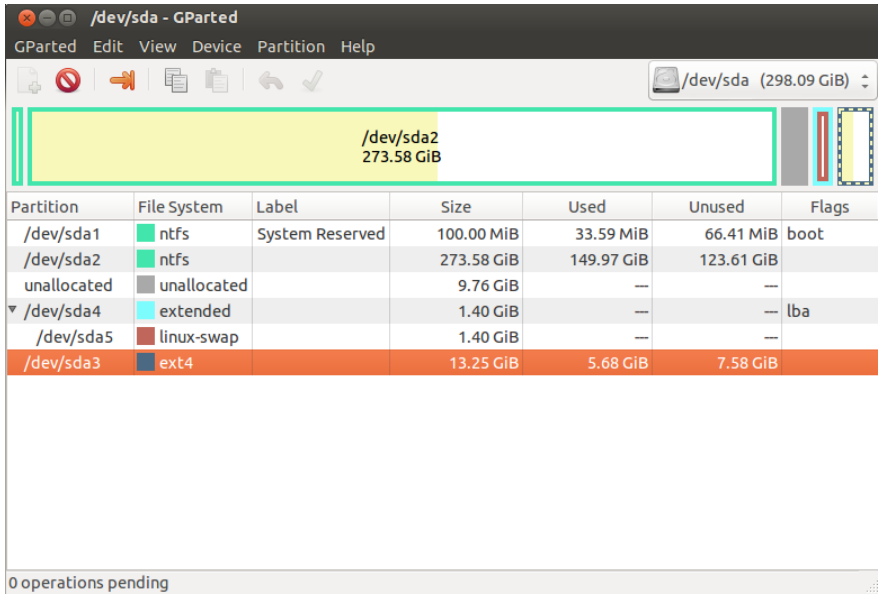
Finding some critical information before moving on

Now, we need one bit of information before we go any further—where Ubuntu actually installed to before things broke. We need this for the next step, and it’s just as easy to get it now before we move on.

Start up a terminal by clicking Show Applications at the bottom left of the screen, typing terminal, and selecting the icon.



- Type "sudo gparted", without the quotes, which is asking to run a disk management tool as an administrator. You can find out more about sudo here, but basically, it gives the command you're running more permissions on your system. Use wisely!
- Gparted will show you what your disks look like. We want to find the one with the Ext4 partition on it. It will look something like the screen below:



There's two important things we need to note here, which we use next:

- The device that the partition is on, which is /dev/sda in the above screenshot
- The partition that hosts the ext4 file system, which is /dev/sda3

Shut down your PC, but leave the USB in it for now.

. . .

Fixing your new installation, and, what’s GRUB?

Ok, so there’s something interesting going on here. At a high level, when you start Ubuntu, you have two things going on

- There’s a boot loader, called GRUB, which helps the system start up.
- Then you have Ubuntu, the applications and desktop, as you’d expect.

In the previous step, GRUB failed to install—so, we still have a full installation of Ubuntu on the PC, but no way of starting it up. This would normally be pretty fatal, but since we have a live USB, we can work around it.

What we’re going to do now is sort of like jump-starting a car. We’ll start up the PC with the USB again, but where it would normally boot through to the live desktop again, we’re going to change some settings.

Startup using the USB disk and hack grub.

As per the first round, you want to boot the PC and select the Ubuntu USB as the startup disk.

When you get to the grub menu, highlight the "Try Ubuntu without installing" entry and press E

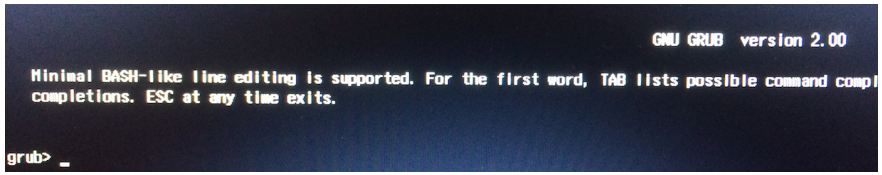
This shows you the config, there's a couple of items of interest here:

```
linux /casper/vmlinuz.efi file=/cdrom/preseed/ubuntu.seed
boot=casper quiet splash ---
```

```
initrd /casper/initrd.lz
```

These are a the commands that configure the boot process in GRUB before the system starts up. We're going to highjack these and point towards the hard disk that we just set Ubuntu up on.

- Exit this menu by pressing escape.
- Press C for the Grub command line, which will drop you back to a prompt:



Now, enter these commands in sequence:

```
linux /casper/vmlinuz.efi root=/dev/<partition>
```

```
initrd /casper/initrd.lz
```

```
boot
```

For example, with the Lenovo, this looks like:

```
linux /casper/vmlinuz.efi root=/dev/mmcblk1p2
```

```
linux /casper/initrd.lz
```

```
boot
```

What we’ve done there is launch your actual, installed copy of Ubuntu as if everything was working OK. This is great, as once we’re in there, we can fix this up and make the change permanent, eg—we won’t need the USB to boot anymore.

. . .

Tidying up grub once you’re in your real Ubuntu

Ok, so now we’re on the final step—we’ve made it back to our actual desktop (you should have had to login after the boot command above), and we can set things straight and ditch the USB.

At this point you want to make sure you’re on WiFi as we need to pull some additional packages in and install them to make sure the system

can run by itself.

Update Apt.

First up, we’re going to run the following command in the terminal:

```
sudo apt-get update
```

This basically updates the library of software that your PC knows about —which helps with the next step. It can take a while so hang tight.

Fix up GRUB install.

Now, we’re going to install / update some packages related to the 32-bit UEFI:

```
sudo apt-get install grub-efi-ia32-bin
```

Then, we’ll tell grub to finish setting itself up properly against the drive on your PC where Ubuntu is installed. Remember earlier when we noted both the Disk and Partition info—we used the partition, now we want the Disk:

```
sudo grub-install /dev/<yourDisk>
```

```
sudo grub-update
```

Again, in the case of the Lenovo this looks like:

```
sudo grub-install /dev/mmcblk1
```

```
sudo grub-update
```

. . .

Voila!

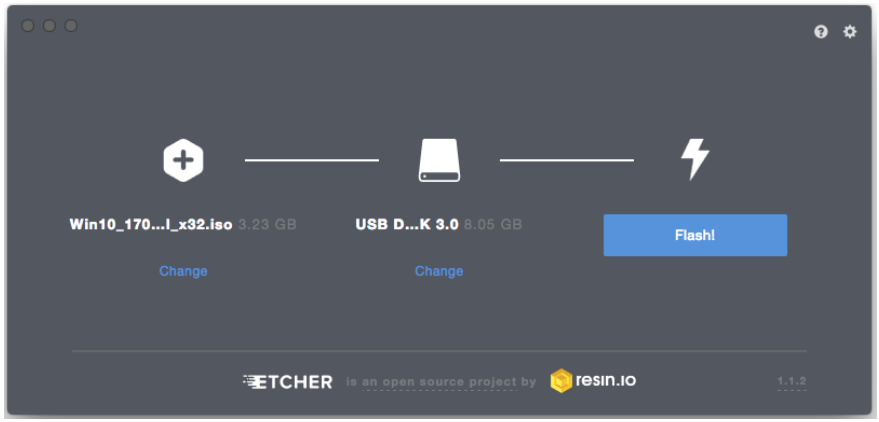
The world being a kind place, now you can remove the USB and restart your machine and work through the normal boot process to arrive back at the login screen.

. . .

Addendum—Reinstalling windows.

Ok, so as promised, if everything goes horribly wrong or you run out of patience, here is the process for getting back to Windows again.

- The windows ISO’s can be downloaded here, and if you’ve been following this guide you should get the 32 bit ISO.
- Grab a copy of Etcher—it’s great for this kind of thing (swear I’m unaffiliated).
- Select the Windows 10 ISO you just downloaded. Etcher may warn about the process to make the drive bootable—it’s ok to ignore this for the EFI installs.
- Kick the process off and let it complete.



- Once it’s ready, kick off the same process as you did for the initial Ubuntu install—booting up your PC and selecting the USB.
- Start the installer and follow the prompts!

Most modern Windows 10 laptops have the key baked into the UEFI, or you can typically find it on the underside of the laptop.