Format and partition hard drive ubuntu

1 Open the terminal. You can open the terminal from dash or press Ctrl+Alt+T. 2 Install GParted. To install GParted, enter the following command: You will be prompted for your user's password, which will not appear as you type: sudo apt-get install gparted Press Y if you are prompted to continue. 3 Start GParted from Dash. Open the dash and type gparted to find the GParted Partition Editor. You'll see a bar that represents the sections on the current volume and their free space. 4 Select the drive you want to format. To select the drive you want to format, click the drop-down menu in the upper-right corner. If you're not sure which one is, use the size of the drive to help you determine. 5 Unmount the section you want to change or remove. You must disconnect the partition before you can make any changes to GParted. Right-click the section from the list or chart, and then select Unmount. 6 Delete the existing partition. This will delete the partition and turn it into unallocated space. You can then create a new partition from this room and format it with the file system. Right-click the section you want to remove, and then click Delete. 7 Create a new partition. After deleting the partition, right-click the unallocated space and select New. This triggers the process of creating a new partition. 8 Select the size of the partition. When you create a new partition, you can use the slider to choose how much space you want to use for it. 9 Select the file system for the partition. Use the File System menu to select the format of the partition. If you plan to use the drive for multiple operating systems and devices, select fat32. If you're just using a drive on Linux, select ext4. 10 Give the partition a leak. This allows you to easily identify it in your system. 11 When you are finished configuring the partition, click Add. The section is added to the order of operations at the bottom of the screen. 12 Resize partition (optional). One of Gparted's features is the ability to resize partitions. You can resize the partition so that a new partition can be created from the resulting free space. This allows you to essentially divide one volume into several pieces. This does not affect the data on the drive. Right-click the section you want to resize, and then select Resize/Move. Drag the edges of the section to create free space before or after it. Click Resize/Move to accept the changes. You need to create new sections from the unallocated space by following the steps above. 13 Click the green check box to start implementing your changes. None of your changes will be applied to the volume until you click this button. When you click it, all sections that you are set to delete will be removed, and you will lose all data. Before continuing, make sure you have the correct settings. All actions may take time to complete, especially if you perform multiple or more actions. is high. 14 Locate your newly formatted drive. When the formatting process is complete, you can close GParted and find your drive. It appears in the list of program drives for files. [2] (Not such installation. lol)I have a brand new 4-TB hard drive to add to my computer recently. So I need to install it on Linux. (Sik-Ho Tsang @ Medium) To do this, I must do the following:A. Partitioning using Parted (> 2GB Harddisk) (New) (Updated 18 January 2020)B. FormattingC. Installation (including auto mount after reboot)D. Checking whether the hard drive is installed. Unmounting (Updated May 05, 2020)F. Partitioning Using fdisk (< 2GB Hard Drive) (old)To partition hard drive larger than 2GB, GPT is needed, > B > C > D.To partition Harddisk less than 2GB, fdisk already can help, E > B > C > D.First, after connecting the hard drive to your computer sata and power cables, we can check the new 4-TB hard drive by:/dev/sdb is a new hard driveNote: You may have a name other than SDB!!! The name sdb must be changed in the following steps!!! 2. But we can not install it now, if we install it now, errors will come out. We need a partition for this first, we use parted partition:3. Within the parted, type the following on the gpt partition, gpt may allow a partition larger than 2GB:4. Set the size of the partition, I here partition 0GB 4GB:5. Then close parted:Format newly partitioned hard disk:Normally the drive is connected to /mnt/. Create a new directory /mnt/ first.2.Then we can install it: 3. But we have to install it every time we restart. Install it automatically after each reboot, I use nano to edit the file / etc / fstab:4. At the end of the file, enter the following: The first item is the path of the hard disk. The second is the destination of the installed drive where we want to install. The third is the format type. On the sixth one I just kept the default, 0 and 0.To check whether the drive SDB, is installed? Use mount command: Subtract, simply using the umount command: But sometimes the installed hard disk may still be in use, which makes you unable to disconnect. Subtract in this situation, is lazy mode: F. Partitioning Using fdisk (<2GB Harddisk) (Old) First, after connecting the hard drive to your computer sata and power cables, we can check the new 4-TB hard drive by:/dev/sdb on a new hard drive2. But we can't install it now, if we install it now, there will be bugs. We must separate it first:3. When we enter m help, we'll see the command list.4. To check the partition table, enter p.5. Partiton, enter n. Then just choose the primary when entering p. And enter 1 only one partition number.6. Insert the w partition table to write to disk. On most computer systems, Linux, or otherwise, when you connect a USB flash drive, you've warned that the drive exists. If the drive is already distributed and formatted to your liking, you just need your computer to list a drive somewhere in your file manager window Desktop. This is a simple requirement and one that the computer generally performs. Sometimes, however, the drive is not set up the way you want. In these times, you need to know how to find and make a storage device connected to your device. What are block devices? A hard disk is commonly called a block device because hard drives read and write data on fixed-size blocks. This distinguishes the hard disk from anything else you can connect to your computer, such as a printer, game control, microphone, or camera. An easy way to list the blocking devices connected to the Linux system is to use the lsblk (list block devices) command: $ lsblk NAME MAJ: min RM SIZE RO TYPE MOUNTPOINT sda 8:00 238.5G 0 disk ... ❤sda1 8:1 0 1G 0 part / boot └ - sda2 8:2 0 237.5G 0 part └ lux-e2bb ... e9f8 253:0 0 237.5G 0 crypt crypt └ ... 253:3 0 181.7G 0 lvm /home sdb 8:16 1 14.6G 0 disc └ -sdb1 8:17 1 14.6G 0 Part Device identifiers are listed in the left column, each starting with SD, and ends with a letter starting with a letter. Each partition on a volume is assigned a number that starts with the number 1. For example, the second partition of the first volume is sda2. If you're not sure what the section is, it's OK - just keep reading. The Lsblk command is non-destructive and is only used for probing, so you can run it without fear of destroying the data from the drive. Testing dmesg In case of doubt, you can test the device label settings by looking at the tail end of the dmesg command, which displays recent system log entries, including kernel events, such as attaching and removing a drive. For example, if you want to make sure that the thumb drive is really / dev / sdc, connect the drive to your computer and run it with the dmesg command: $ sudo dmesg | Tail The last ride listed is the one you just connected. If you disconnect it from the mains and restart this command, you'll see that the device has been removed. If you reconnect it and run the command, the device will be there. In other words, you can monitor the kernel awareness of your car. Understanding file systems If all you need is a device tag, your job is done. But if your goal is to create a drive that you use, you must provide the file system to the drive. If you're not sure what a file system is, it's probably easier to understand the concept by learning what happens when you don't have a file system at all. If you have a free drive that doesn't have any important data about it, you can follow this example. Otherwise, do not try this exercise, as it will definitely delete the data provided. It is possible to use the drive without a file system. If you are definitely correctly identified by the drive, and you have completely confirmed there is nothing significant, connect it to your computer-but don't install it. If it automatically mounts, disconnect it manually. $ su - # umount /dev/sdx{,1} To protect against catastrophic copy-pasting errors, these examples are used sdx tag. Now that the drive is disconnected, try this: # echo hello world > / dev / sdx You have just written data to block the device without it being installed on your system or having a file system. To get the data you just wrote, you can check the raw data to drive: #head -n 1 / dev / sdx hello world that seemed to work pretty well, but imagine that the phrase hello world is one file. If you want to write a new file using this method, you need: Know, there is already a file in line 1 Know that an existing file takes only 1 line To infer a way to add new data, or else write line 1 while writing line 2 For example: # echo hello world > this is another file >> /dev/sdx to get the first file, nothing changes. # head -n 1 /dev/sdx hello world But it is more difficult to get another file. # head -n 2 /dev/sdx | tail -n 1 this is another file Probably this method of writing and reading data is not practical, so developers have created systems to monitor what constitutes a file where one file starts and ends, and so on. Most file systems require a partition. Creating partitions A hard disk partition is a kind of border on your device that tells each file system what space it can occupy. For example, if you have a 4GB thumb drive, you may have a partition that the device takes the entire drive (4GB), two partitions that each take 2GB (or 1 and 3 if you want), three some variation sizes, and so on. The combinations are almost endless. Assuming that your drive is 4GB, you can create one large partition from the terminal whose GNU command is structured: # parted /dev/sdx --align opt mklabel msdos 0 4G This command first specifies the path of the device, as requires parted. The --align option allows the parted to find the optimal starting and stopping point for the partition. The mklabel command creates a partition table (called a disk label) on the device. In this example, msdos uses the tag because it is a very compatible and popular label, although gpt is becoming more common. The starting and ending points in the section are last defined. Because the --align opt flag is used, the parted drive adjusts the size needed to optimize performance, but these numbers are a guide. Then, create an actual partition. If your start and end options aren't optimal, parted will warn you and ask if you want to make adjustments. # parted /dev/sdx -a opt mkpart primary 0 4G Warning: received partition not properly aligned with best performance: 1s % 2048s != 0s Ignore / Cancel? C # parted /dev/sdx -a opt mkpart primary 2048s 4G When you run lsblk again (you need to remove the drive and connect it back), you will see that your drive now has one partition on it. Many file systems are available for manual creation of the file system. Some are free and open source, while others are not. Some companies refuse to support open source file systems so that their users cannot read open file systems, while open source users cannot read closed without reverse engineering of them. This disconnect, regardless of whether there are many file systems you can use, and the one you choose depends on the drive purpose. If you want the drive to be compatible on many systems, your only choice is currently with the exFAT file system. Microsoft has not provided exFAT code to any open source kernel, so you may need to install exFAT support with your package manager, but exFAT support is included in both Windows and MacOS. If you have exFAT support installed, you can create an exFAT file system on your drive in the partition you created. # mkfs.exfat -n myExFatDrive /dev/sdx1 Now your drive is readable and writable for closed systems and open source systems using additional (and as yet unpunished Microsoft) kernel modules. A common file system common on Linux is ext4. This is arguably a cumbersome file system for portable drives because it retains user rights that are often different from one computer to another, but it is generally a reliable and flexible file system. As long as you're comfortable managing permissions, ext4 is a great, magazine file system for portable hard drives. # mkfs.ext4 -L myExt4Drive /dev/sdx1 Connect your drive and reconnect it. For Ext4 portable drives, use sudo to create a catalog and to give permission for this catalog to the user and group that are common to your systems. If you're not sure which user and group to use, you can change read and write permissions by using a sudo or root system that is having problems with the drive. Using desktop tools It's great to know how to deal with hard drives for nothing but a Linux kest standing on your and block the device, but sometimes you just want to get the drive ready for use without so much insightful testing. Excellent tools from both GNOME and KDE developers can make it easy to prepare a drive. GNOME Disks and KDE Partition Manager are graphical interfaces that offer everything this article has explained so far. Run one of these apps to see a list of added devices (left column), create or resize partitions, and create a file system. GNOME version is expected to be easier than the KDE version, so I demo a more complex one – it's easy to understand GNOME Disks if that's what you have handy. Start THE CDE partition manager and enter your root password. In the left column, select the disk that you want to format. If your drive isn't listed, make sure it's connected, then select Tools, > Update devices (or F5 on your keyboard). Do not continue unless you are ready to destroy the existing partition table on the drive. When the drive is selected, click New Partition Table on the top toolbar. You will be prompted to select the tag you want to give to the partition table: either gpt or msdos. The former is more flexible and able to cope with larger disks, while the latter, like many Microsoft technologies, are de facto the force of standard market share. Now that you have fresh table, right-click your device in the right panel and select New to create a new partition. Follow the instructions to specify your section type and size. This action combines the partitioning step with creating a file system. To apply your changes to the drive, click Apply in the upper-left corner of the window. Hard drives, easy hard drives Dealing with hard drives are easy for Linux, and it's even easier if you understand the language of hard drives. Since switching to Linux, I've been better equipped to prepare hard drives in any way I want them to work for me. It has also been easier for me to recover lost data because of the transparency Linux offers when it comes to storage. Here are some recent tips for testing and more information about hard disks: Back up your data, not just the data on the drive you're testing. All it takes is one wrong move to destroy the partition with an important drive (which is a great way to learn about recreating lost partitions, but not much fun). Check and then check again to see if your targeted drive is the right drive. I often use lsblk to make sure I haven't driven with myself. (It's easy to remove two drives from two separate USB ports, then frantically attach them in the second order, causing them to get new drive labels.) Take the time to destroy the test drive and see if you can recover the data. This is a good learning experience to recreate a partition table or to get data back after you remove the file system. If you have a closed operating system, try working on an open source file system. There are some projects working towards this kind of compatibility, and trying to get them working in a stable and reliable way is a good weekend project. Page 2: On most computer systems, on Linux, or in other ways, when you connect a USB flash drive, you have warned that the drive exists. If the drive is already distributed and formatted to your liking, you just need your computer to list the drive somewhere in the File Manager window or desktop. This is a simple requirement and one that the computer generally performs. Sometimes, however, the drive is not set up the way you want. In these times, you need to know how to find and make a storage device connected to your device. What are block devices? A hard disk is commonly called a block device because hard drives read and write data on fixed-size blocks. This distinguishes the hard disk from anything else you can connect to your computer, such as a printer, game control, microphone, or camera. An easy way to list the blocking devices connected to the Linux system is to use the lsblk (list block devices) command: $ lsblk NAME MAJ: min RM SIZE RO TYPE MOUNTPOINT sda 8:00 238.5G 0 disk ... ❤sda1 8:1 0 1G 0 part / boot └ - sda2 8:2 0 237.5G 0 part └ lux-e2bb ... e9f8 253:0 0 237.5G 0 crypt crypt ... fedora-root 253:1 0 50G 0 LVM / ^ fedora-swap 253:2 0 5.8G 0 LVM [SWAP] └ 0 181.7G 0 LVM /home sdb 8:16 1 14.6G 0 disk └ - sdb1 8:17 1 114.6G 0 part device identifiers are listed in the left column, each starting with sd, and ending with a letter starting. Each partition on a volume is assigned a number that starts with the number 1. For example, the second partition of the first volume is sda2. If you're not sure what the section is, it's OK - just keep reading. The Lsblk command is non-destructive and is only used for probing, so you can run it without fear of destroying the data from the drive. Testing dmesg In case of doubt, you can test the device label settings by looking at the tail end of the dmesg command, which displays recent system log entries, including kernel events, such as attaching and removing a drive. For example, if you want to make sure that the thumb drive is really / dev / sdc, connect the drive to your computer and run it with the dmesg command: $ sudo dmesg | Tail The last ride listed is the one you just connected. If you disconnect it from the mains and restart this command, you'll see that the device has been removed. If you reconnect it and run the command, the device will be there. In other words, you can monitor the kernel awareness of your car. Understanding file systems If all you need is a device tag, your job is done. But if your goal is to create a drive that you use, you must provide the file system to the drive. If you're not sure what a file system is, it's probably easier to understand the concept by learning what happens when you don't have a file system at all. If you have a free drive that doesn't have any important data about it, you can follow this example. Otherwise, do not try this exercise, as it will definitely delete the data provided. It is possible to use the drive without a file system. If you are definitely correctly identified by the drive, and you have completely confirmed there is nothing significant, connect it to your computer-but don't install it. If it automatically mounts, disconnect it manually. $su - #umount /dev/sdx{,1} To protect against catastrophic copy-pasting errors, these examples use an unlikely sdx tag for the drive. Now that the drive is disconnected, try this: # echo hello world > / dev / sdx You have just written data to block the device without it being installed on your system or having a file system. To get the data you just wrote, you can check the raw data to drive: #head -n 1 / dev / sdx hello world that seemed to work pretty well, but imagine that the phrase hello world is one file. If you want to write a new file using this method, you need: Know, there is already a file in line 1 Know that an existing file takes only 1 line To infer a way to add new data, or else write line 1 while writing line 2 For example: # echo hello world > this is another file >> /dev/sdx to get the first file, nothing changes. # head -n 1 /dev/sdx hello world But it is more difficult to get another file. # head -n 2 /dev/sdx | tail -n 1 this is another file Probably this writing and reading data is not practical, so developers have created systems to monitor what constitutes a file where one file starts and ends, and so on. Most file systems require a partition. Creating partitions A hard disk partition is a kind of border on your device that tells each file system what space it can occupy. For example, if you have a 4GB thumb drive, you may have a partition that the device takes the entire drive (4GB), two partitions that each take 2GB (or 1 and 3 if you want), three some variation sizes, and so on. The combinations are almost endless. Assuming that your drive is 4GB, you can create one large partition from the terminal whose GNU command is structured: # parted /dev/sdx --align opt mklabel msdos 0 4G This command first specifies the path of the device, as requires parted. The --align option allows the parted to find the optimal starting and stopping point for the partition. The mklabel command creates a partition table (called a disk label) on the device. In this example, msdos uses the tag because it is a very compatible and popular label, although gpt is becoming more common. The starting and ending points in the section are last defined. Because the --align opt flag is used, the parted drive adjusts the size needed to optimize performance, but these numbers are a guide. Then, create an actual partition. If your start and end options aren't optimal, parted will warn you and ask if you want to make adjustments. # parted /dev/sdx -a opt mkpart primary 0 4G Warning: received partition not properly aligned with best performance: 1s % 2048s != 0s Ignore / Cancel? C # parted /dev/sdx -a opt mkpart primary 2048s 4G When you run lsblk again (you need to remove the drive and connect it back), you will see that your drive now has one partition on it. Many file systems are available for manual creation of the file system. Some are free and open source, while others are not. Some companies refuse to support open source file systems so that their users cannot read from open file systems, while open source users cannot read closed files without reverse engineering. This disconnect, regardless of whether there are many file systems you can use, and the one you choose depends on the drive purpose. If you want the drive to be compatible on many systems, your only choice is currently with the exFAT file system. Microsoft has not provided exFAT code to any open source kernel, so you may need to install exFAT support with your package manager, but exFAT support is included in both Windows and MacOS. If you have exFAT support installed, you can create an exFAT file system on your drive in the partition you created. # mkfs.exfat -n myExFatDrive /dev/sdx1 Now your drive is readable and writable for closed systems and open source systems using additional (and as yet unpunished Microsoft) kernel modules. A common file system common on Linux is ext4. This is undeniably cumbersome file system with portable drives it retains user rights that are often different from one computer to another, but it is generally a reliable and flexible file system. As long as you're comfortable managing permissions, ext4 is a great, magazine file system for portable hard drives. # mkfs.ext4 -L myExt4Drive /dev/sdx1 Connect your drive and reconnect it. For Ext4 portable drives, use sudo to create a catalog and to give permission for this catalog to the user and group that are common to your systems. If you're not sure which user and group to use, you can change read and write permissions by using a sudo or root system that is having problems with the drive. Using desktop tools It's great to know how to deal with hard drives for nothing but a Linux kest standing on your and block the device, but sometimes you just want to get the drive ready for use without so much insightful testing. Excellent tools from both GNOME and KDE developers can make it easy to prepare a drive. GNOME Disks and KDE Partition Manager are graphical interfaces that offer everything this article has explained so far. Run one of these apps to see a list of added devices (left column), create or resize partitions, and create a file system. GNOME version is expected to be easier than the KDE version, so I demo a more complex one – it's easy to understand GNOME Disks if that's what you have handy. Start THE CDE partition manager and enter your root password. In the left column, select the disk that you want to format. If your drive isn't listed, make sure it's connected, then select Tools, > Update devices (or F5 on your keyboard). Do not continue unless you are ready to destroy the existing partition table on the drive. When the drive is selected, click New Partition Table on the top toolbar. You will be prompted to select the tag you want to give to the partition table: either gpt or msdos. The former is more flexible and able to cope with larger disks, while the latter, like many Microsoft technologies, are de facto the force of standard market share. Now, if you have a fresh partition table, right-click the right panel on your device and select New to create a new partition. Follow the instructions to specify your section type and size. This action combines the partitioning step with creating a file system. To apply your changes to the drive, click Apply in the upper-left corner of the window. Hard drives, easy hard drives Dealing with hard drives are easy for Linux, and it's even easier if you understand the language of hard drives. Since switching to Linux, I've been better equipped to prepare hard drives in any way I want them to work for me. It has also been easier for me to recover lost data because of the transparency Linux offers when it comes to storage. Here are some recent tips for testing and more information about hard disks: Back up your data, not just the data on the drive you're testing. All it takes is one wrong step to destroy. partition important drive (which is a great way to learn about recreating lost partitions, but not much fun). Check and then check again to see if your targeted drive is the right drive. I often use lsblk to make sure I haven't driven with myself. (It's easy to remove two drives from two separate USB ports, then frantically attach them in the second order, causing them to get new drive labels.) Take the time to destroy the test drive and see if you can recover the data. This is a good learning experience to recreate a partition table or to get data back after you remove the file system. If you have a closed operating system, try working on an open source file system. There are some projects working towards this kind of compatibility, and trying to get them working in a stable and reliable way is a good weekend project. Project.