

Backing up and restoring operating systems

- Introduction -
- Backing up your OS -
- Restoring your backup -
- Making your backups bootable -
- Booting OSES with other kernels -
- Summary -

Introduction

Have you ever lost your OS install due to breaking it and being unable to fix?

Have you ever lost your OS install due to drive failure?

Have you ever lost your OS install due to a theft or a fire?

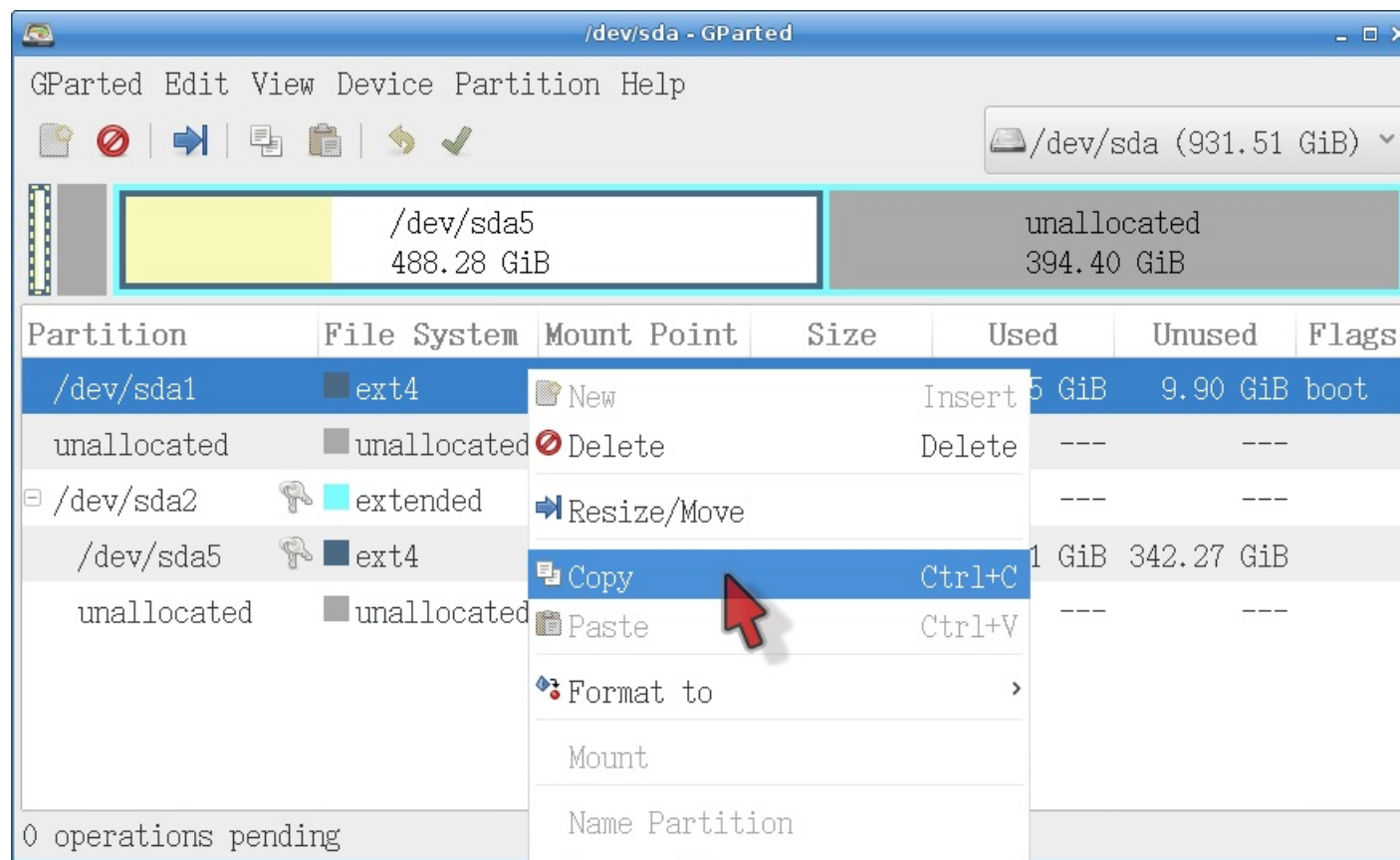
Have you ever wanted to copy your OS install to another computer without reinstalling it and redoing all the modifications to the base distro?

Well, hopefully you haven't suffered from the first three. But if you have, you know the value of backups. And if you haven't, well, let me tell you that you don't want those things to happen.

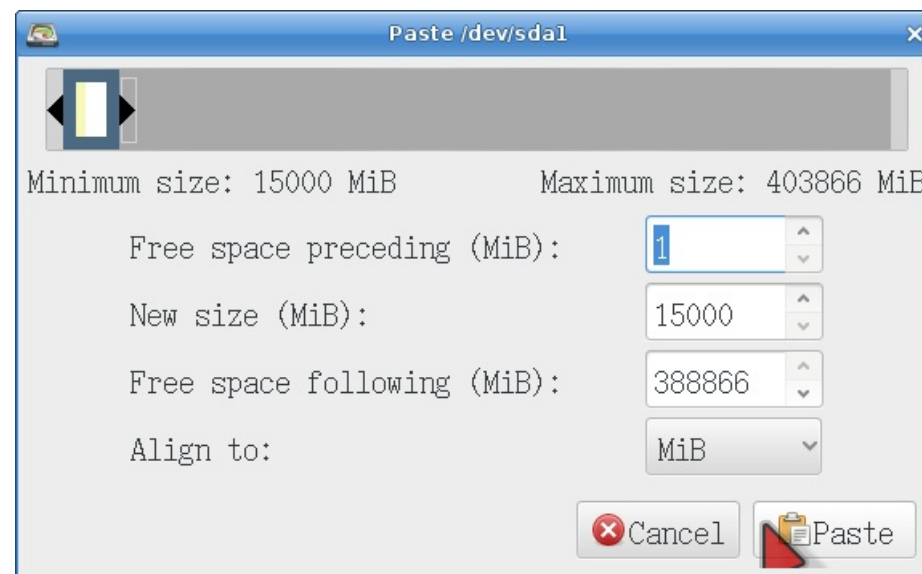
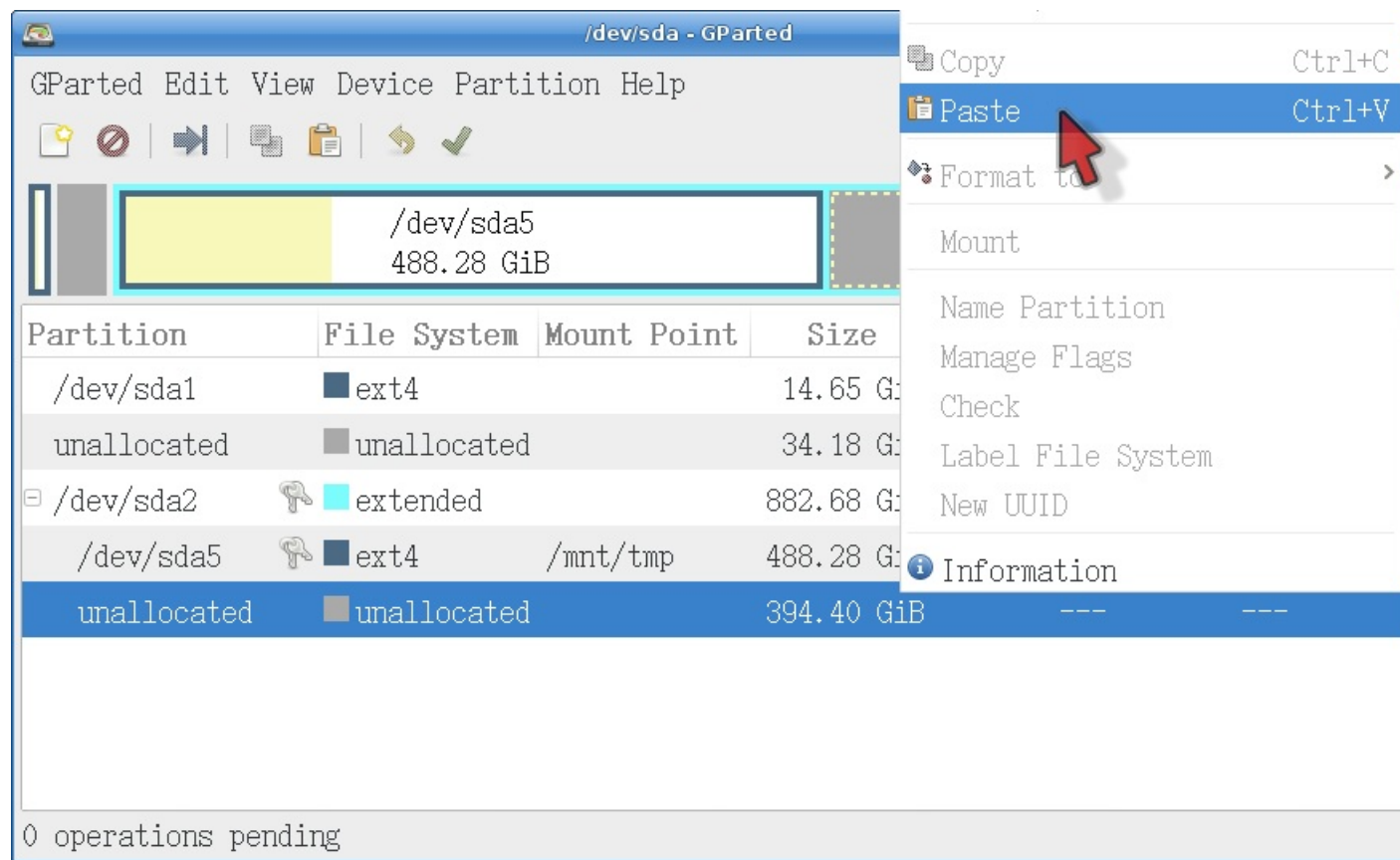
As for the fourth - if you are anything like me, you probably have your favorite, carefully prepared system configuration that you don't stray away from (except to check stuff out sometimes, maybe). Wouldn't it be great if you could just "freeze" and "teleport" your OS whenever and wherever you want to? Protecting you both from those listed hazards and saving time / effort redoing modifications in case you want to deploy your OS elsewhere. This article will teach you those magic spells. What we will do here is make a snapshot of your current running OS and store it on other partitions and / or drives - and then learn how to bring it back. The main reason for doing this is to immunize you against OS breaks; you can always play around with your current install and just restore the backup in case of a break. Anyway let's go:

Backing up your OS

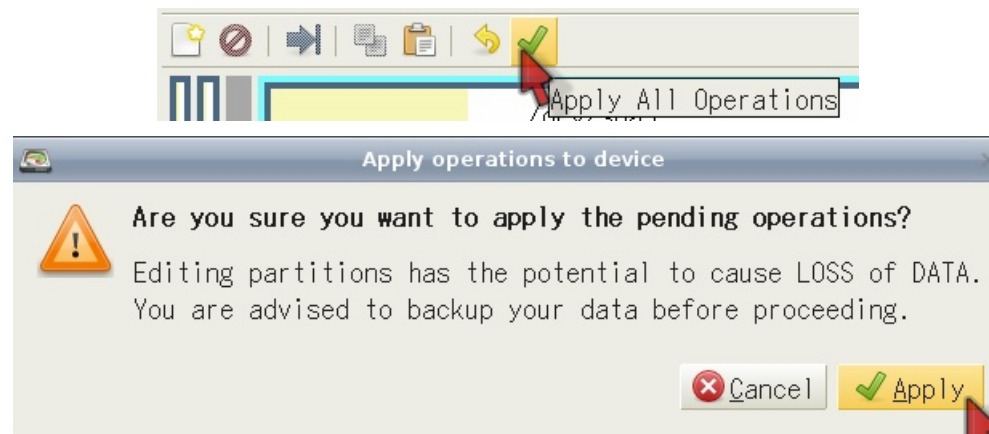
1. Log in to another Linux installation (on the same computer). This might be an OS that's already installed on another partition / drive, or even a live USB disk.
2. Launch GParted
3. Right click and copy the partition that has the system you want to backup (it is easier to do this entire process if your /home is on the same partition - in that case, all your settings will be backed up, as well. But not strictly essential):



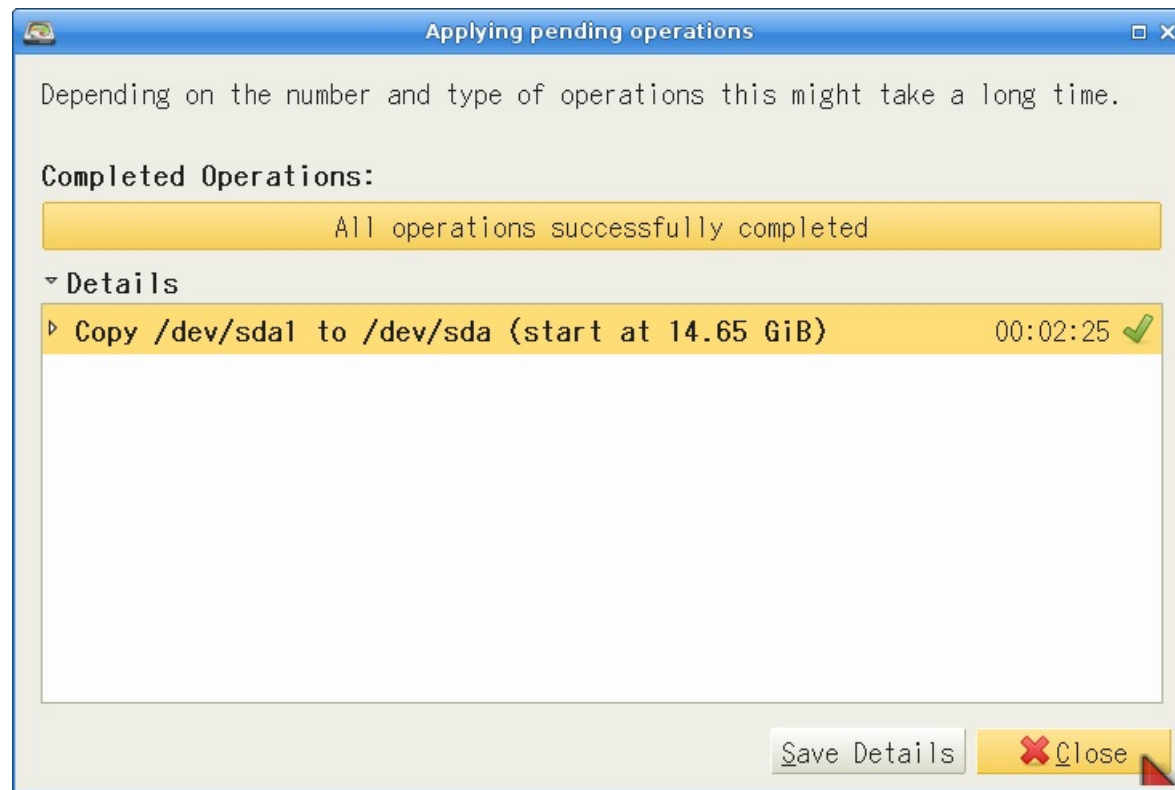
4. Find an amount of unallocated space that is higher than the size of your copied partition (if you don't have, you are going to need to shrink and / or delete partitions), right click and paste:



5. Proceed with the operation:

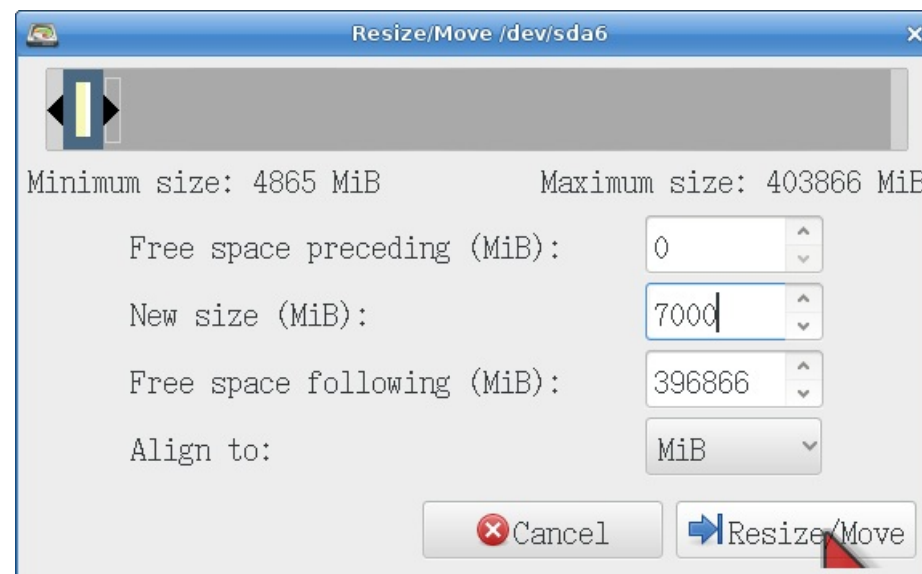
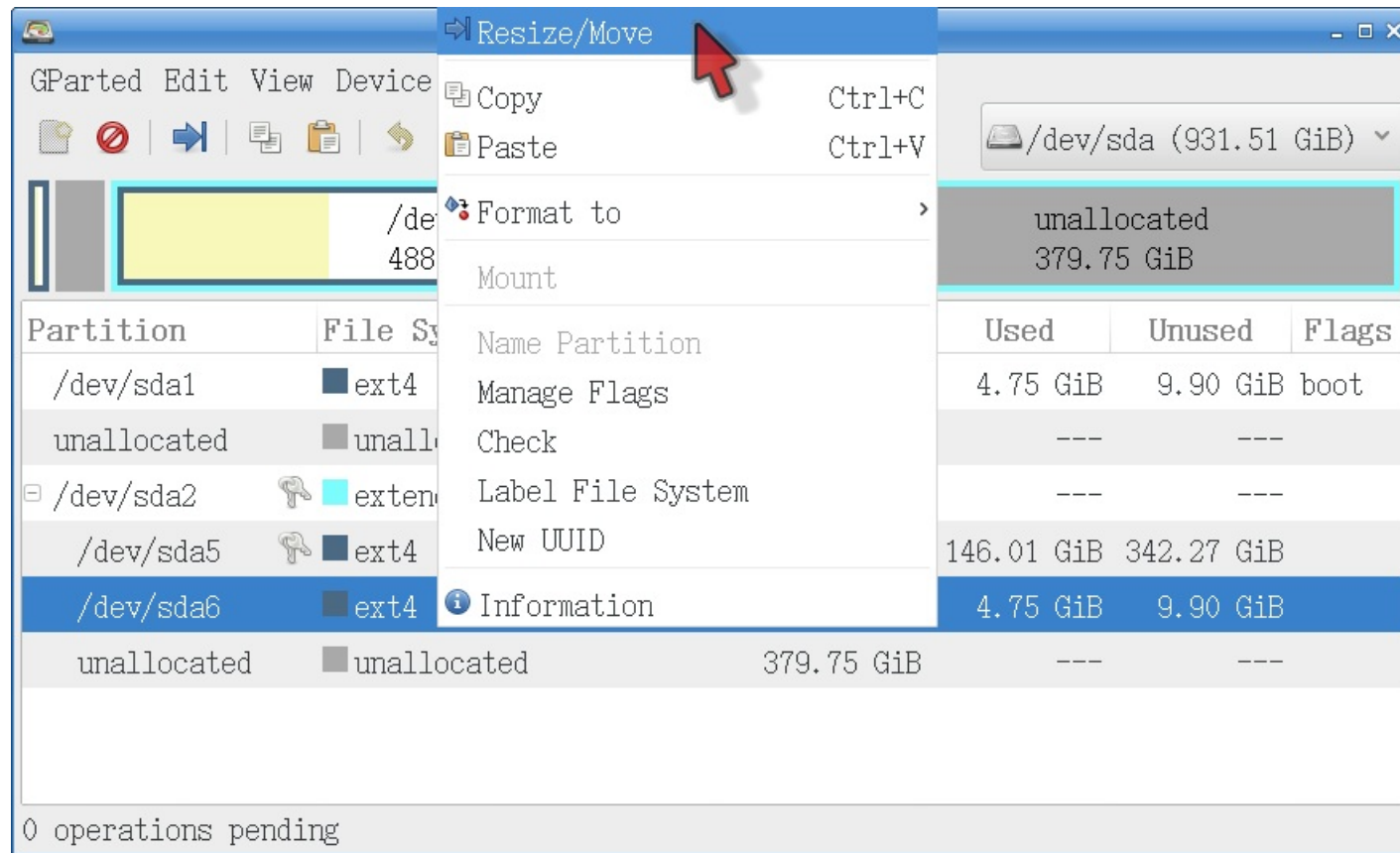


6. Wait until it finishes, and close:



7. Now, shrink that copied partition (**not the actual one!**) to the minimum possible space. Though, you will need a little more than what GParted says is the minimum - so if you have a 7GB system, try 9GB just to be safe (if there's too little, it will result in the *“New size*

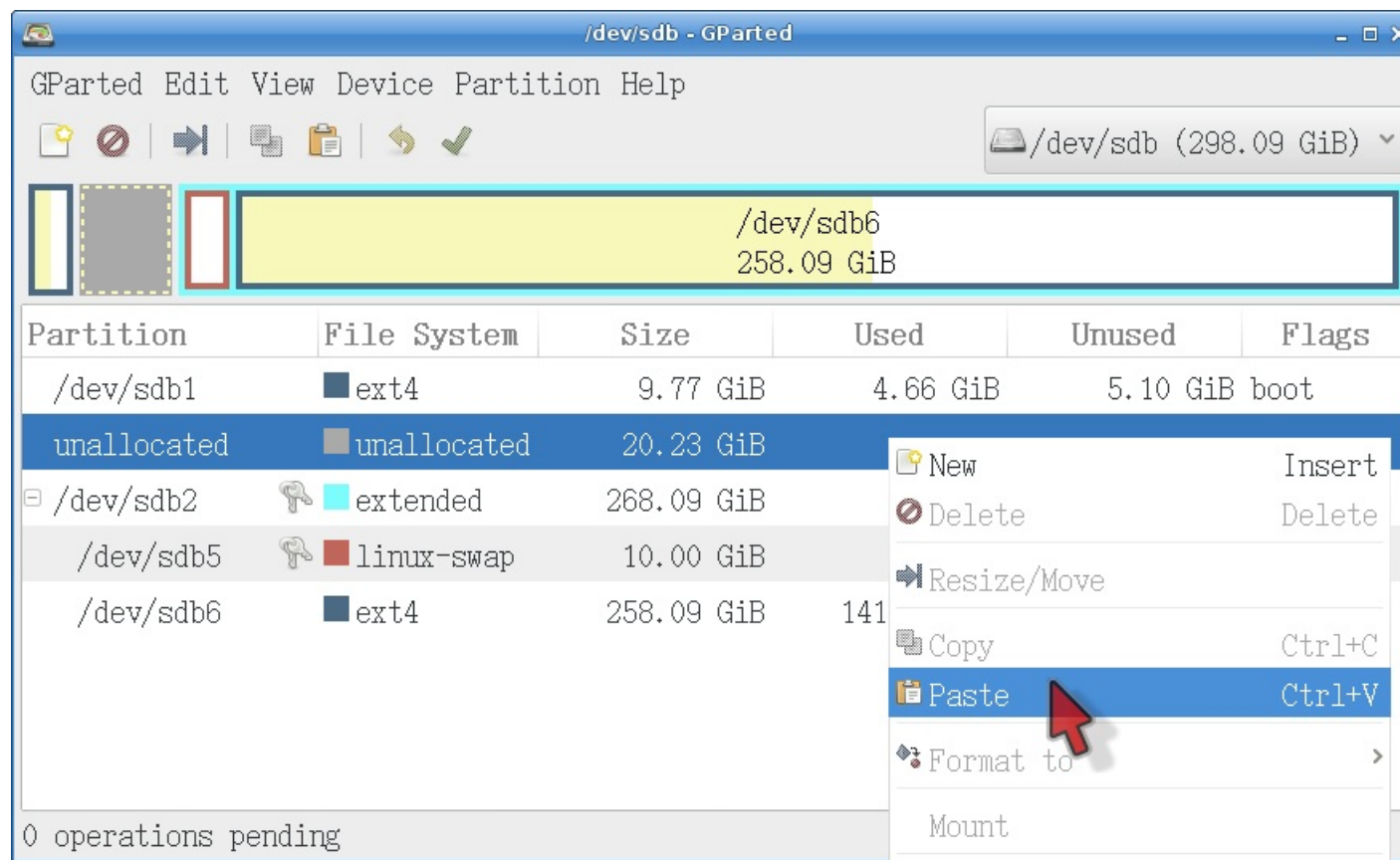
smaller than the minimum” error):



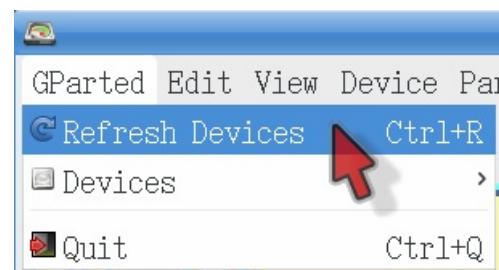
8. Apply and confirm the operation as in step 5. What the shrinking accomplishes is that your OS backup will fit into less space later (e.g a smaller pendrive). If you want to use the *dd* command to restore it later, it will also take much less time, then.
9. Copy the newly made partition as in step 3 (except a different identifier this time - *sda6* in our screenshots).
10. We will make yet another backup - this time on another drive. This way - if your first drive dies and takes with it your backup - you still have another. Switch your drive (you **do** have more than one, right?):



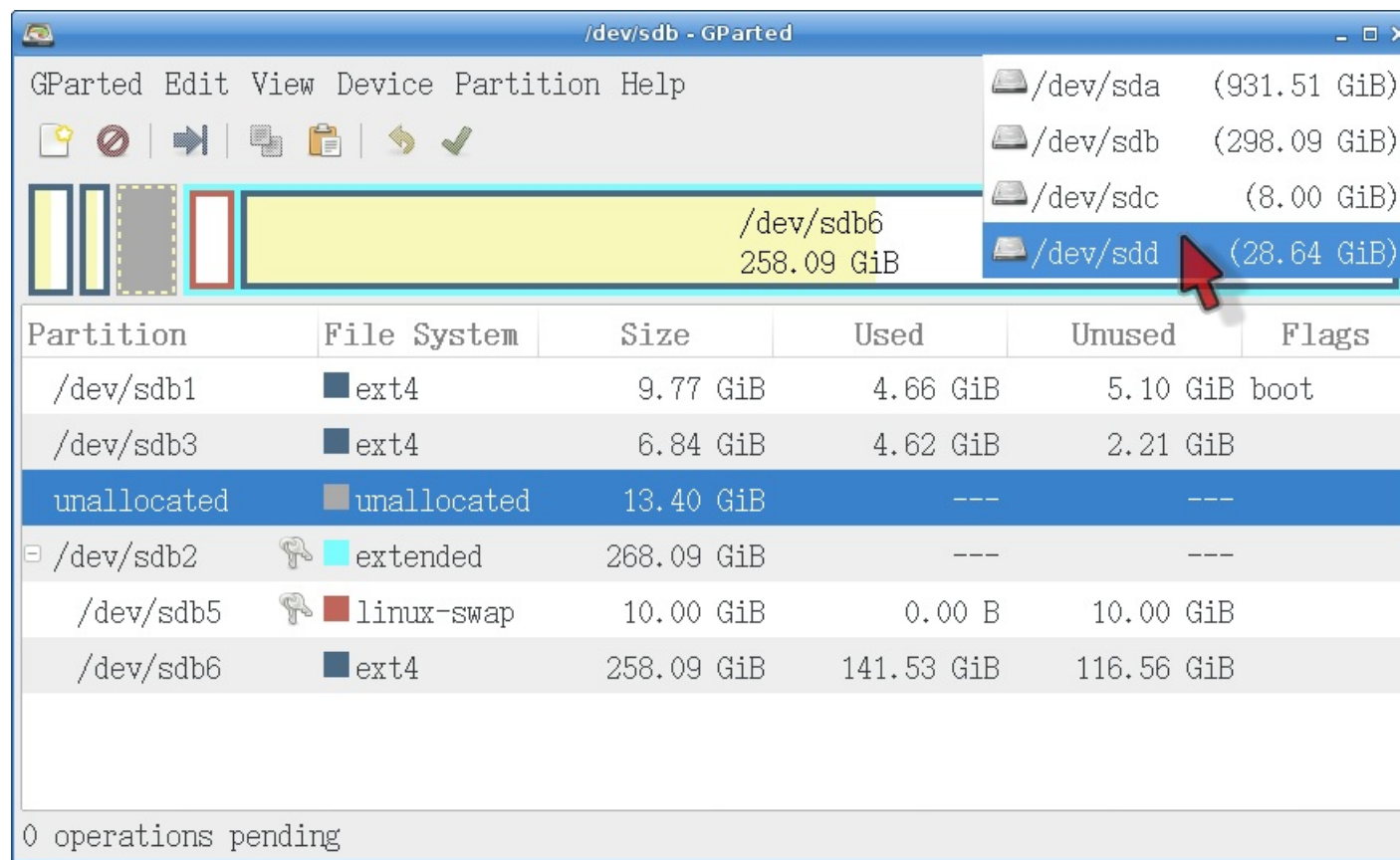
11. Paste the partition in unallocated space again (just on the other drive):



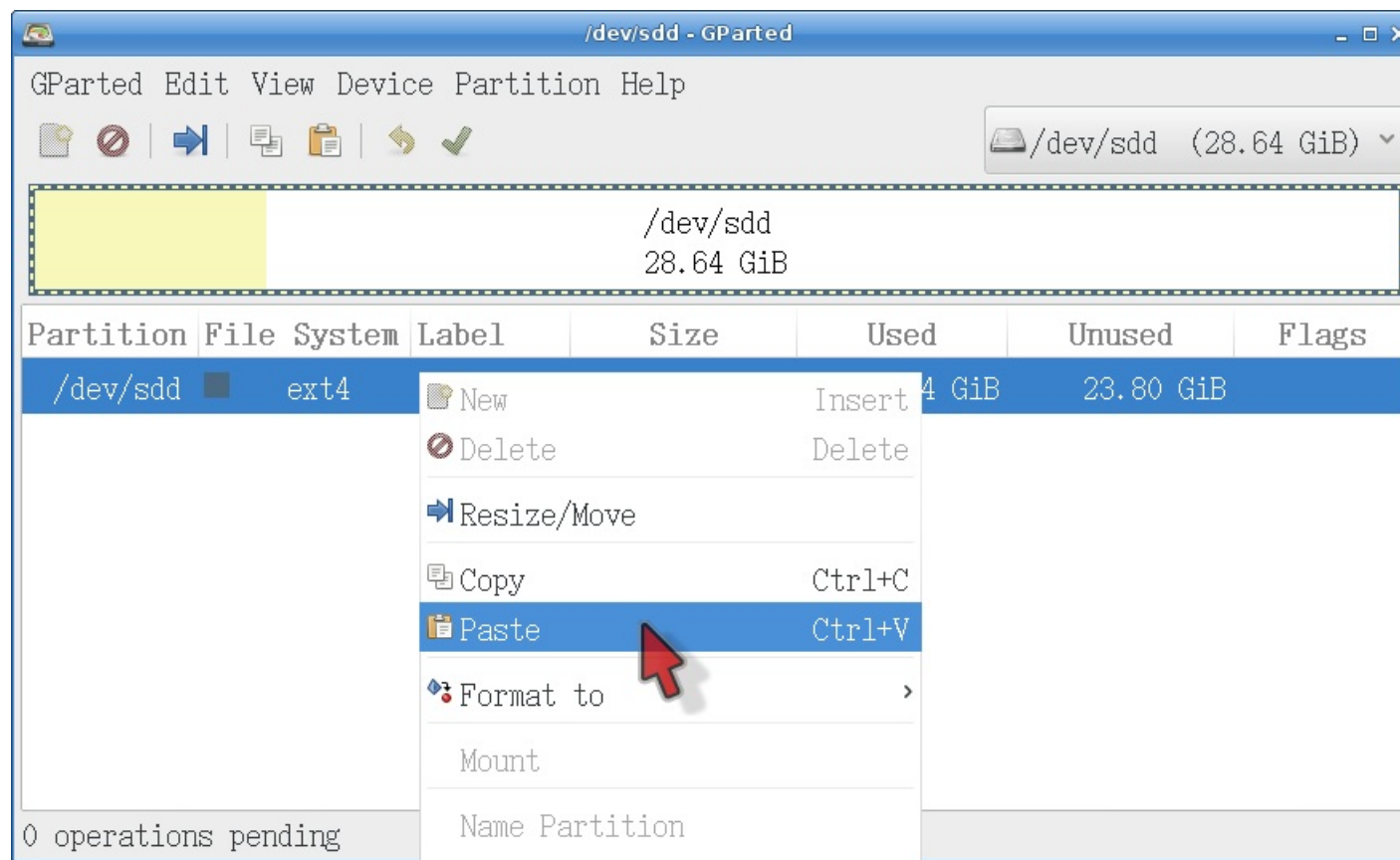
12. Apply and confirm as usual. Now attach a pendrive and refresh devices:



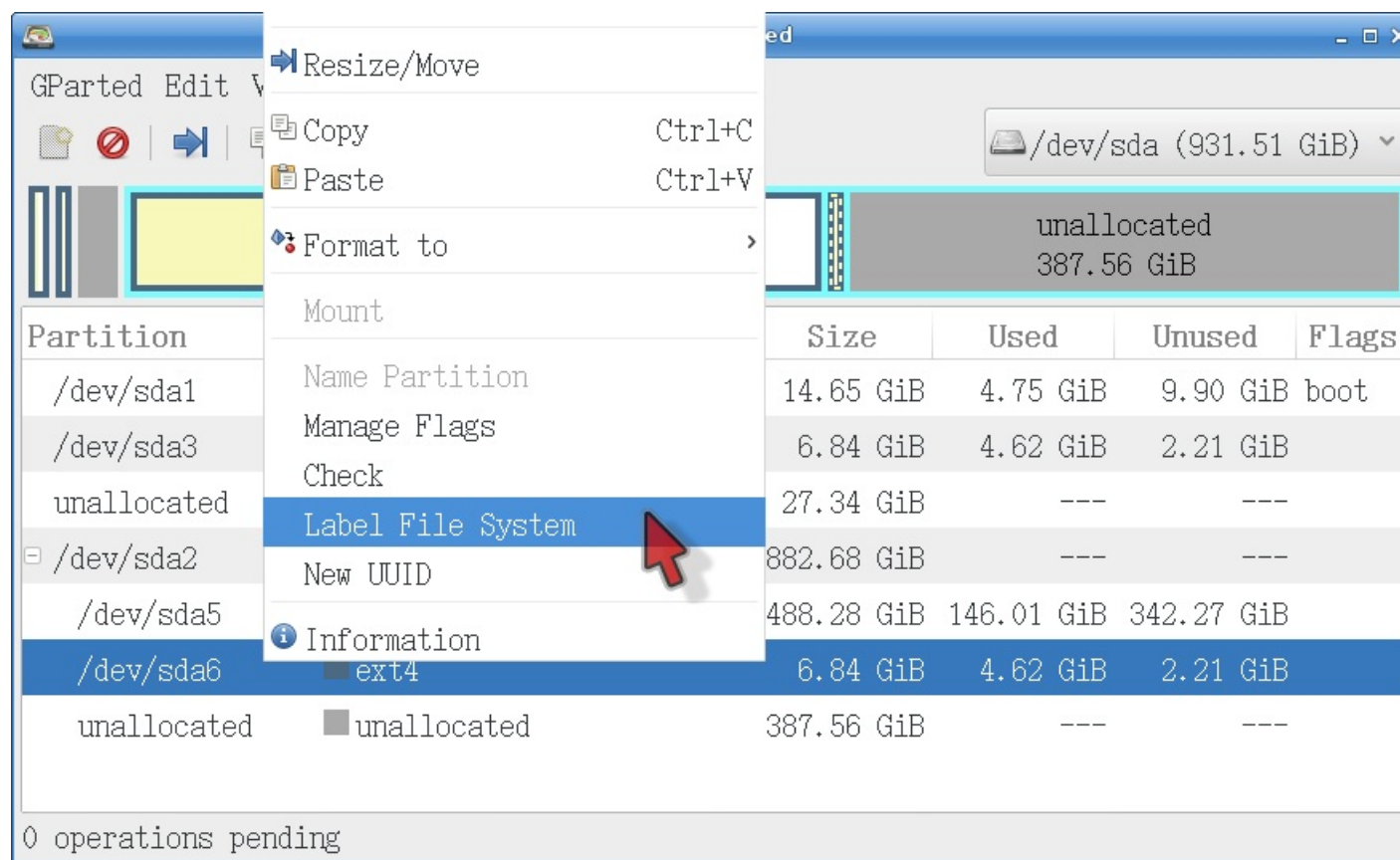
13. A new device will appear. Switch to it (this will be the one with the "furthest" letter in the alphabet):



14. Copy and paste your OS again, this time onto the pendrive:



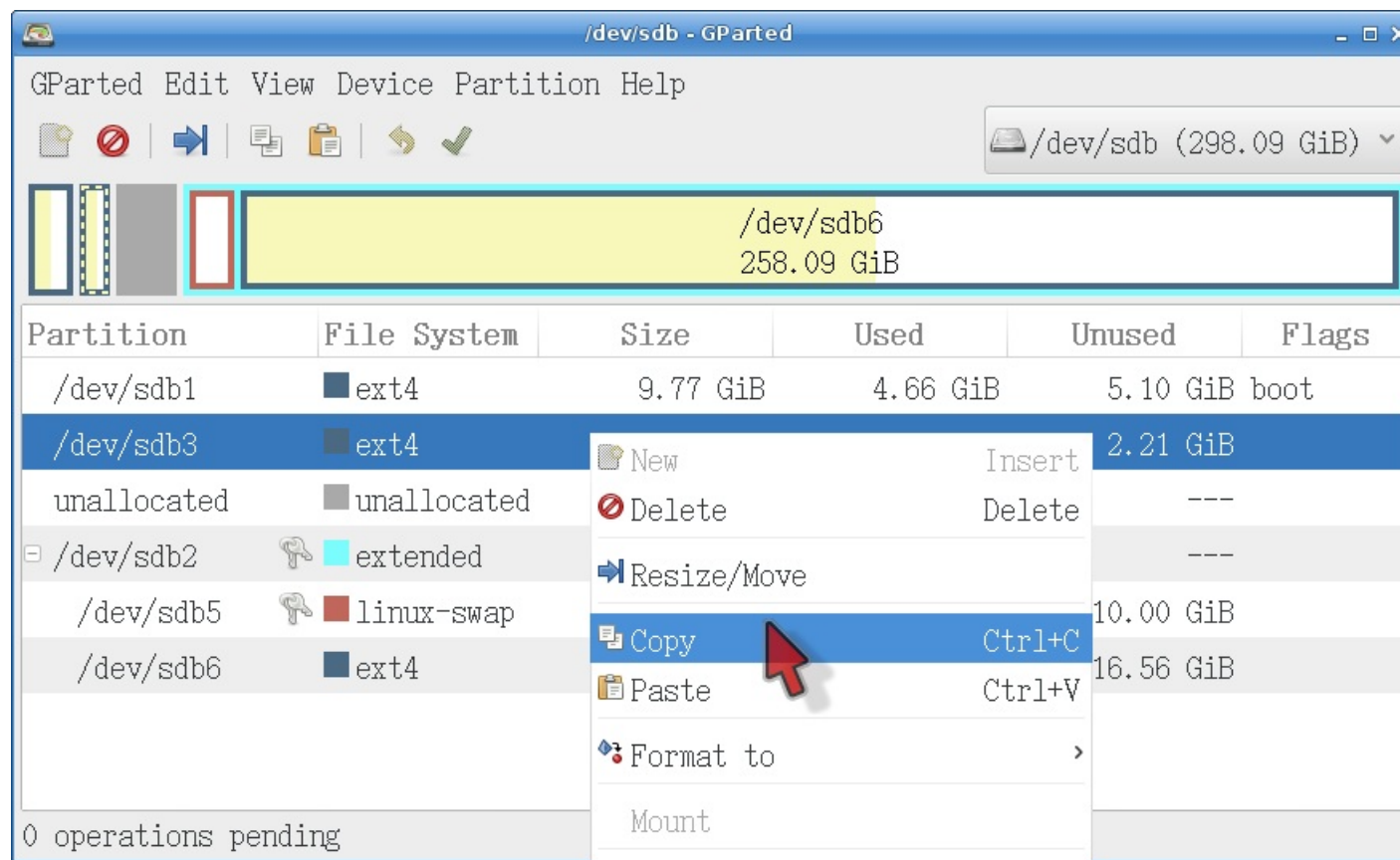
15. Apply and confirm as usual. Now, you can stick that pendrive backup onto a relative or a friend in case of a fire or theft in your house. Remember to label your backups, so that it is easier to recognize them:



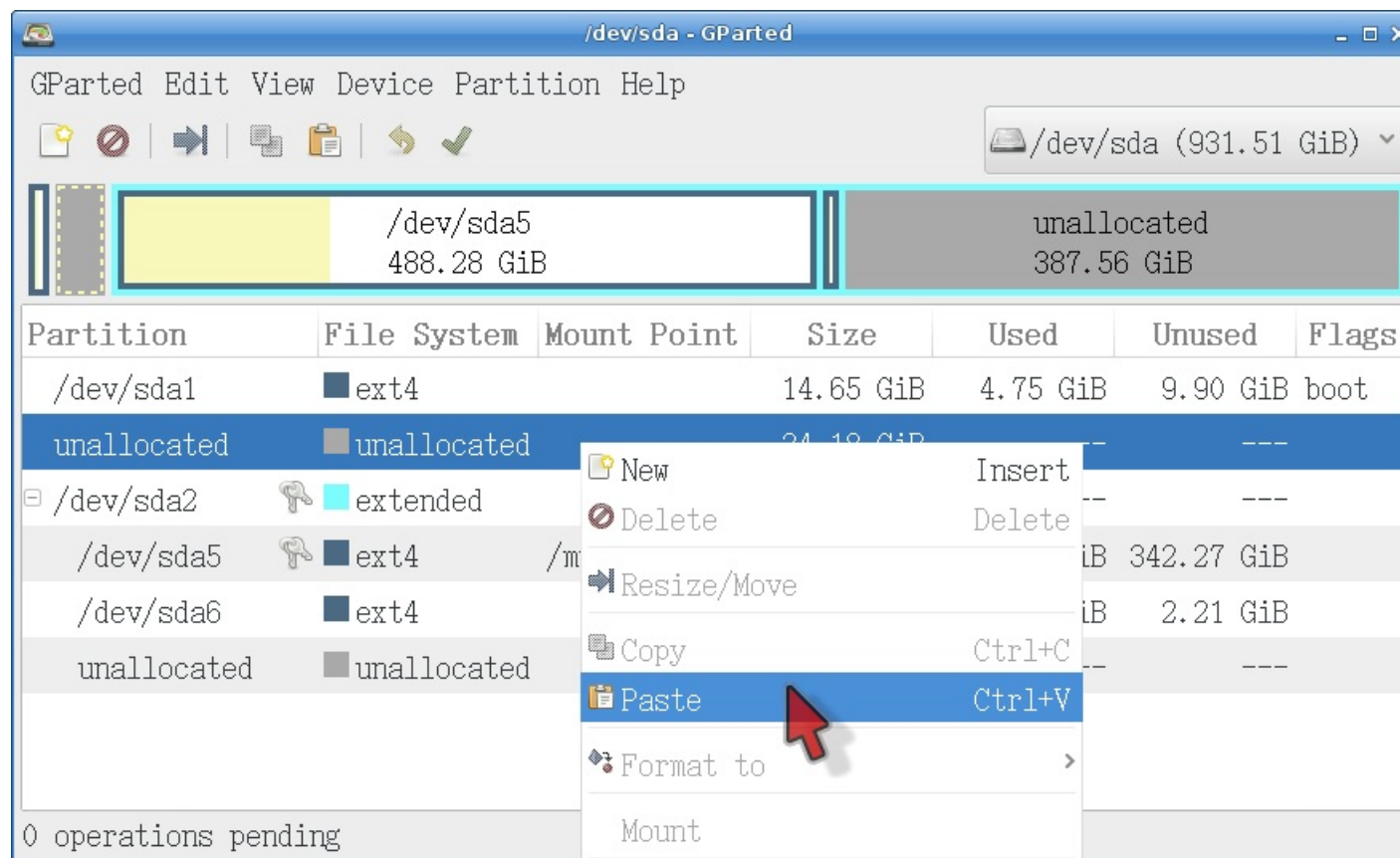
16. Do that for all of the backups.

Restoring your backup

1. To restore a backup, we will redo the entire process but in reverse. So copy the backup:



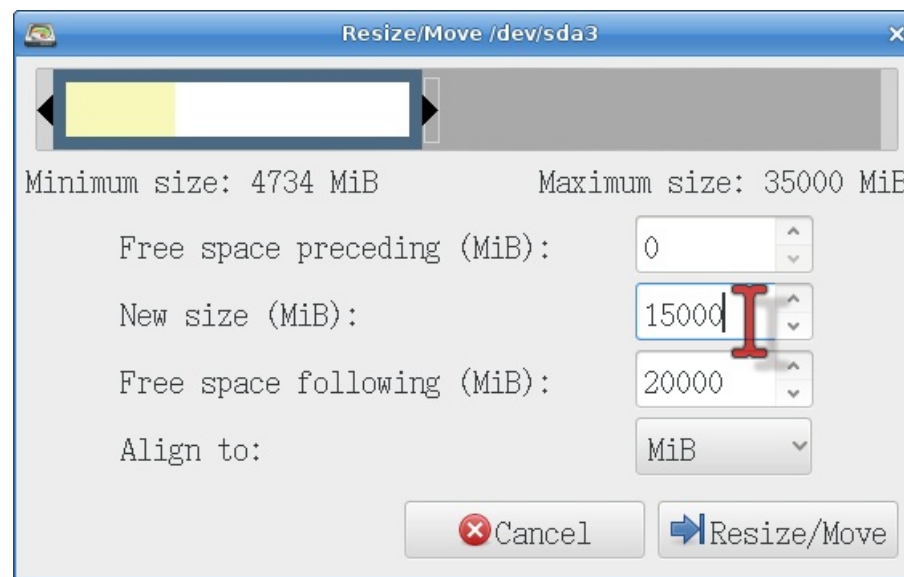
2. Switch your drive to whichever you want to restore the backup to. Paste to unallocated space:



3. Apply and confirm as usual. Here is our restored backup:

Partition	File System	Mount Point	Size	Used	Unused	Flags
/dev/sda1	ext4		14.65 GiB	4.75 GiB	9.90 GiB	boot
/dev/sda3	ext4		6.84 GiB	4.62 GiB	2.21 GiB	

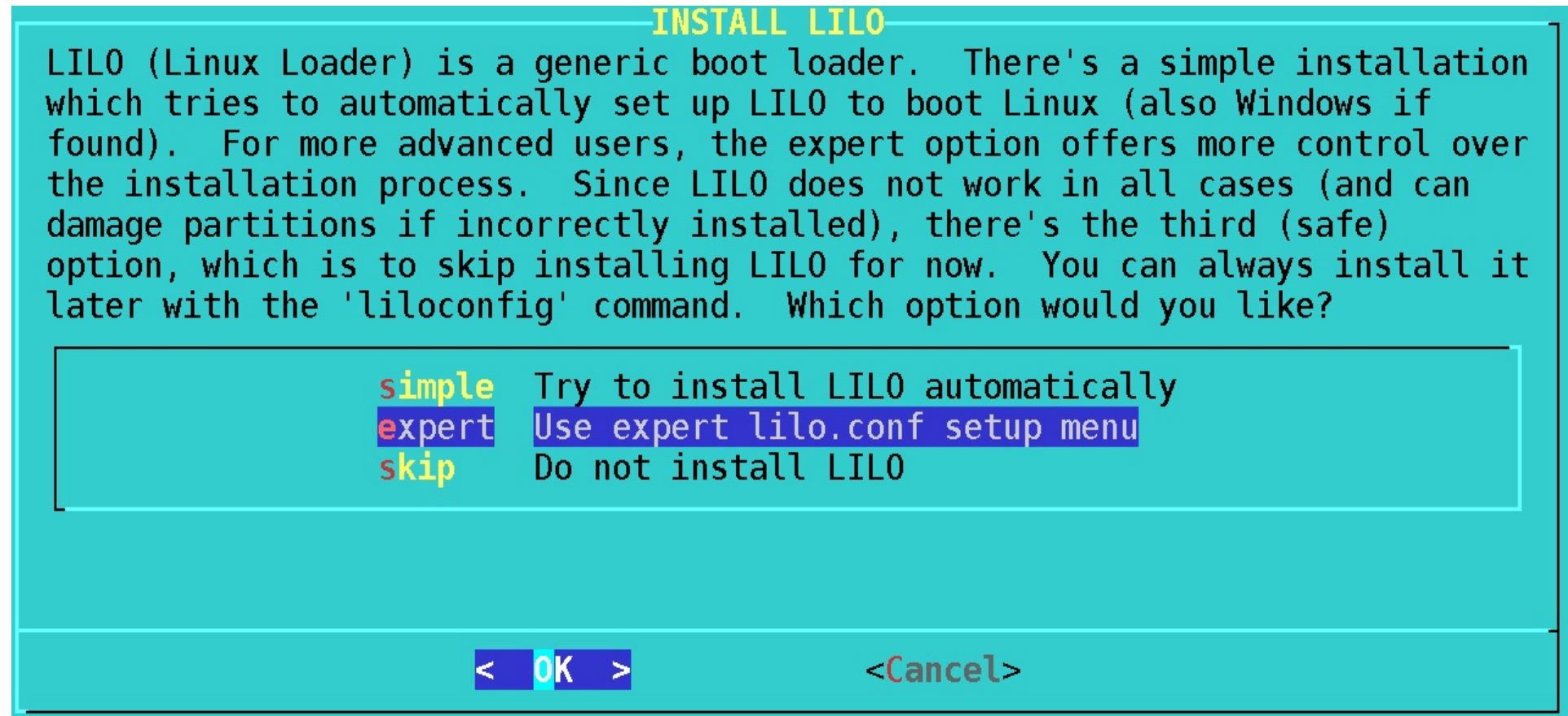
4. We're going to need to enlarge it now (since presumably, you will still install stuff on it after that). Click “*Resize/Move*” and input your desired size:



Making your backups bootable

UPDATE February 2025: the Lilo included by default in Slackware 15 might not work for this, and you might need to downgrade (or rather upgrade in this case) to the one from 14.2 for example. A few weeks ago I tried to setup a few bootable OSes in my drive(s), but whatever I did with the configs, drive positions, etc. was failing - Lilo could still only boot from `/dev/sda1` regardless. I thought it might have been because of some modifications I've done to my usual OS, so I tried installing Lilo from a pure Slackware 15, which did nothing either. At the final sign of desperation I thought to try the last remaining option, which was to use another version of Lilo - even though the Internet had nothing about a "bug" in the new version. And voila! It fixed the problem. I never noticed it earlier because I was almost always booting `/dev/sda1`; and even when I was trying another option and it failed, I just assumed I had setup Lilo incorrectly and didn't think too much of it. But no; there is probably some combination of old hardware (like my junkyard one) that activates this "bug" in Lilo. Keep this in mind before you read the following part of the tutorial.

1. To boot your backups, you're going to have to inform your bootloader about them. Run *“sudo liloconfig”* and pick the *“expert”* mode. Note: this entire process will assume you're using a live system that runs the same kernel as your restored backups; if it's not the case, the following instructions are still valid, but will need a tiny modification after you're done.



2. Now pick *“Begin”*:

EXPERT LILO INSTALLATION

This menu directs the creation of the LILO config file, lilo.conf. To install, you make a new LILO configuration file by creating a new header and then adding one or more bootable partitions to the file. Once you've done this, you can select the install option. Alternately, if you already have an /etc/lilo.conf, you may reinstall using that. If you make a mistake, you can always start over by choosing 'Begin'. Which option would you like?

Begin	Start LILO configuration with a new LILO header
Linux	Add a Linux partition to the LILO config
Windows	Add a Windows FAT or NTFS partition to the LILO config
Install	Install LILO
Recycle	Reinstall LILO using the existing lilo.conf
Skip	Skip LILO installation and exit this menu
View	View your current /etc/lilo.conf
Help	Read the Linux Loader HELP file

< OK >

<Cancel>

3. The next screen should not be needed, so just “OK” past it:

Some systems might require extra parameters to be passed to the kernel. If you needed to pass parameters to the kernel when you booted the Salix bootdisk, you'll probably want to enter the same ones here. Most systems won't require any extra parameters. If you want to use hibernation, please add a resume option for your swap partition. For example: 'resume=/dev/sda3' if your swap partition is /dev/sda3. If you don't need any, just hit ENTER to continue.

< OK >

<Cancel>

4. Just pick “*standard*” on the next screen:

CONFIGURE LILO TO USE FRAME BUFFER CONSOLE?

Looking at /proc/devices, it seems your kernel has support for the Linux frame buffer console. If we enable this in /etc/lilo.conf, it will allow more rows and columns of text on the screen and give you a cool penguin logo at boot time. However, the frame buffer text console is slower than a standard text console. In addition, not every video card or monitor supports all of these video modes. Would you like to use the frame buffer console, or the standard Linux console?

s standard	Use the standard Linux console (the safe choice)
640x480x64k	Frame buffer console, 640x480x64k
800x600x64k	Frame buffer console, 800x600x64k
1024x768x64k	Frame buffer console, 1024x768x64k
1280x1024x64k	Frame buffer console, 1280x1024x64k
1600x1200x64k	Frame buffer console, 1600x1200x64k

< OK >

<Cancel>

5. Pick “*MBR*” now:

SELECT LILO TARGET LOCATION

LILO can be installed to a variety of places: the superblock of your root Linux partition (which could then be made the bootable partition with fdisk), a formatted floppy disk, or the master boot record of your first hard drive. If you're using a boot system such as Boot Manager, you should use the Root selection. Please pick a target location:

Root	Install to superblock (not for use with XFS)
Floppy	Use a formatted floppy disk in the boot drive
MBR	Use the Master Boot Record (possibly unsafe)

< OK >

<Cancel>

6. Going to assume you've picked the first drive, but you can change this to */dev/sdb* or any other drive you want to install the bootloader to:

CONFIRM LOCATION TO INSTALL LILO

The auto-detected location to install the LILO boot block is shown below. If you need to make any changes, you can make them below. Otherwise, hit ENTER to accept the target location shown.

/dev/sda

< OK >

<Cancel>

7. Pick “*30 seconds*” here, so that you can actually manage to choose which OS you want to

boot (main or backup):

CHOOSE LILO TIMEOUT

At boot time, how long would you like LILO to wait for you to select an operating system? If you let LILO time out, it will boot the first OS in the configuration file by default.

None	Don't wait at all - boot straight into the first OS
5	5 seconds
30	30 seconds
Forever	Present a prompt and wait until a choice is made

< OK >**<Cancel>**

8. If using Salix, you will get to choose if you want its logo to display on boot. Just deny it (or allow if you like, whatever):

OPTIONAL SALIX LOGO BOOT SCREEN

Would you like to use a boot screen with the Salix logo against a black background? If you answer no here, the standard LILO menu will be used.

< Yes >**< No >**

9. Now add a bootable system:

EXPERT LILO INSTALLATION

This menu directs the creation of the LILO config file, lilo.conf. To install, you make a new LILO configuration file by creating a new header and then adding one or more bootable partitions to the file. Once you've done this, you can select the install option. Alternately, if you already have an /etc/lilo.conf, you may reinstall using that. If you make a mistake, you can always start over by choosing 'Begin'. Which option would you like?

Begin	Start LILO configuration with a new LILO header
Linux	Add a Linux partition to the LILO config
Windows	Add a Windows FAT or NTFS partition to the LILO config
Install	Install LILO
Recycle	Reinstall LILO using the existing lilo.conf
Skip	Skip LILO installation and exit this menu
View	View your current /etc/lilo.conf
Help	Read the Linux Loader HELP file

< OK >

<Cancel>

10. First, add the main system (*sda1* - which was the origin of all our backups and that you are presumably using regularly):

SELECT LINUX PARTITION --default-item /dev/sdb1

Which Linux partition would you like LILO to boot?

Partition		Start	End	Sectors	ID	
/dev/sda1	*	2048	30722047	30720000	14.7G	83 Linux
/dev/sda3		30722048	45058047	14336000	6.9G	83 Linux
/dev/sda5		102404096	1126404095	1024000000	488.3G	83 Linux
/dev/sda6		1126406144	1140742143	14336000	6.9G	83 Linux
/dev/sdb1	*	2048	20482047	20480000	9.8G	83 Linux
/dev/sdb3		20482048	34818047	14336000	6.9G	83 Linux
/dev/sdb6		83892224	625141759	541249536	258.1G	83 Linux

< OK >

<Cancel>

11. Input a name that will remind you of it and “OK” past the screen:

SELECT PARTITION NAME FOR /dev/sda1

Now you must select a short, unique name for this partition. You'll use this name if you specify a partition to boot at the LILO0 prompt. 'Linux' might not be a bad choice. THIS MUST BE A SINGLE WORD.

Main

< OK >

<Cancel>

12. Add another linux system, this time the backup. Pick “*Linux*” again, and add the *sda3* partition on which the backup is installed:

SELECT LINUX PARTITION --default-item /dev/sdb1

Which Linux partition would you like LILO to boot?

Partition		Start	End	Sectors	ID	
/dev/sda1	*	2048	30722047	30720000	14.7G	83 Linux
/dev/sda3		30722048	45058047	14336000	6.9G	83 Linux
/dev/sda5		102404096	1126404095	1024000000	488.3G	83 Linux
/dev/sda6		1126406144	1140742143	14336000	6.9G	83 Linux
/dev/sdb1	*	2048	20482047	20480000	9.8G	83 Linux
/dev/sdb3		20482048	34818047	14336000	6.9G	83 Linux
/dev/sdb6		83892224	625141759	541249536	258.1G	83 Linux

< OK >

<Cancel>

13. Input the appropriate name:

SELECT PARTITION NAME FOR /dev/sda3

Now you must select a short, unique name for this partition. You'll use this name if you specify a partition to boot at the LILO prompt. 'Linux' might not be a bad choice. THIS MUST BE A SINGLE WORD.

< OK >

<Cancel>

14. Finally install the bootloader:

EXPERT LILO INSTALLATION

This menu directs the creation of the LILO config file, lilo.conf. To install, you make a new LILO configuration file by creating a new header and then adding one or more bootable partitions to the file. Once you've done this, you can select the install option. Alternately, if you already have an /etc/lilo.conf, you may reinstall using that. If you make a mistake, you can always start over by choosing 'Begin'. Which option would you like?

Begin	Start LILO configuration with a new LILO header
Linux	Add a Linux partition to the LILO config
Windows	Add a Windows FAT or NTFS partition to the LILO config
Install	Install LILO
Recycle	Reinstall LILO using the existing lilo.conf
Skip	Skip LILO installation and exit this menu
View	View your current /etc/lilo.conf
Help	Read the Linux Loader HELP file

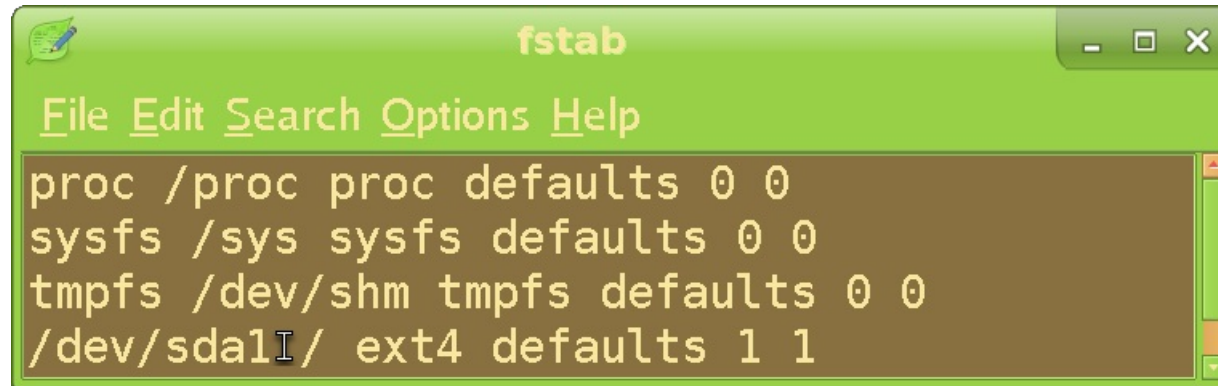
< OK >

<Cancel>

Installing the Linux Loader...

15. Now you will be able to choose between booting your main system and your restored backup if you choose the first drive at the BIOS level (you could have theoretically put as many OSes in there as you wanted to, even ones installed on other drives; we're going to

keep it simple for now, though). Before the backup is operational - we're going to have to point its fstab partitions towards the correct identifiers. Type *“sudo mount /dev/sda3 /mnt/tmp”* in a terminal. Then *“sudo leafpad /mnt/tmp/etc/fstab”*:



```
fstab
File Edit Search Options Help
proc /proc proc defaults 0 0
sysfs /sys sysfs defaults 0 0
tmpfs /dev/shm tmpfs defaults 0 0
/dev/sda1/ ext4 defaults 1 1
```

The fstab file decides from which partition the system is actually booted (and every other partition that will be mounted at launch). See how it's still trying to boot from *sda1*, even though the backup is on *sda3*? That's because the original fstab had sda1 in it - as it should have. But now we're trying to run the restored backup, so this is counterproductive. Just change it to *“sda3”* and save.

Booting OSES with other kernels

Adding a Linux distro in liloconfig's interactive menu will insert an entry pointing the bootloader to the same kernel that the OS you ran liloconfig from has. For example, the lilo config file might have a section like this:

```
# Linux bootable partition config begins
image = /boot/vmlinuz-generic
initrd = /boot/initrd.gz
```

```
root = /dev/sdb1
label = Slack14
read-only # Partitions should be mounted read-only for checking
# Linux bootable partition config ends
```

In Slackware, *vmlinuz-generic* is actually just a symlink to whatever the kernel version it's using, eg. *vmlinuz-generic-5.15.63* in 15. If the OS you ran liloconfig from has that, then lilo will "expect" all the OSes defined in the config file to be using it too. Yet, imagine that you wanted lilo to boot an older or newer Slackware version (like the 14 in the above example), or even a completely different distro. The bootloader will expect the booted OS to be using the kernel specified in the config file, but it doesn't, so it will fail to load. To fix this problem, you will need to **copy the kernel from the other distro to your current /boot folder, and point lilo to it**. First, mount the partition that contains the "other" distro you want to boot (eg “*sudo mount /dev/sdb1 /mnt/tmp*”), and copy its kernel to */boot* (where liloconfig loads them from by default), eg “*sudo cp /mnt/tmp/boot/vmlinuz-generic-4.16.9 /boot*”; remember to pick the one with the numbers at the end. Now modify the lilo config file entry as such:

```
# Linux bootable partition config begins
image = /boot/vmlinuz-generic-4.16.9
initrd = /boot/initrd.gz
root = /dev/sdb1
label = Slack14
read-only # Partitions should be mounted read-only for checking
# Linux bootable partition config ends
```

What was just *vmlinuz-generic* became *vmlinuz-generic* + the kernel version your "different"

OS is using. Save and re-run lilo so that it "absorbs" the updated config file (just *"sudo lilo"* is enough). There is no problem for lilo be able to load many different Slackware version installs, or even different distros, though I have no experience with the latter. Some internet posters are saying that certain distros also have their own unique initrd files; if so, those will have to be copied to */boot* and inserted into the config file, as well.

Summary

That's it! We've restored a backup. You can reboot in peace now. Even though there seem to be a lot of steps, this is actually easier than it seems if you become familiar with it. I use it every time I break shit and it's surely a lot easier than reinstalling a distro and redoing your changes (ugh). TBD: dd command backups.

[Back to the front page](#)