How To Use A Swap File Instead Of A Swap Partition On Linux

This article explains how to transition from having a swap partition to a swap file. If you don't need to disable any existing swap partition and all you need is to create a swap file and activate it, simply skip steps 1 and 2.

On my Ubuntu 18.04 desktop I had a fairly large swap partition which I wanted to use for other purposes, and move the swap to a file. Ubuntu 18.04 already uses a swap file by default instead of a swap partition, however, I upgraded to the latest Ubuntu version instead of making a clean install, so my system continued to use a swap partition. Therefore I had to move the swap to a file myself.

As a result, the instructions below were tested on my Ubuntu 18.04 desktop. They should work on any Linux distribution though.

It's important to mention that you can't use a swap file with a BTRFS filesystem (thanks to Isaac for mentioning this in the comments).

Also, hibernating (to disk) will no longer work out of the box when using a swap file. This can be done but I can't test it because resuming from hibernation didn't work on my system previously to switching to a swapfile so I just gave up on using hibernation. What's more, most Linux distributions use suspend (to RAM) instead of hibernate (to disk) by default anyway. If you need to enable hibernation with a swapfile, there's some info here. Suspend (to ram) is not affected by this.

How to Move Swap To A File On Your Linux Filesystem

1. Turn off your current swap partition.

To see the active swap partition, run:

```
swapon -s
```

The command output looks like this in my case:

```
Filename    Type        Size       Used   Priority
/dev/sda5   partition   15624188   0      -2
```

Now you can turn off the current swap device using this command:

```
sudo swapoff /dev/sdXX
```

Where `/dev/sdXX` is the device listed by the `swapon -s` command (under the Filename section - `/dev/sda5` in my case from the example above), so make sure to replace it with your swap partition.

2. Remove your old swap entry from the /etc/fstab file.

To remove the old swap entry, open the `/etc/fstab` file as root with a text editor, and remove the swap line. Do not modify anything else in the `/etc/fstab` file! Changing anything else in this file may prevent your system from booting!

You can open the file with Nano editor from the command line, like this:

```
sudo nano /etc/fstab
```

And remove the entry containing your swap partition information (you can also just comment out the line by adding a `#` in front of it). As an example, in my case the swap entry looks like this:

```
UUID=d1b17f9c-9c5e-4471-854a-3ccaf358c30b none swap sw 0 0
```

As you can see, the swap entry should contain `swap` and `sw` - that's how you know which line to remove (or comment out).

Then press `Ctrl + O`, then `Enter` to save the file. To exit Nano editor after you've saved the file press `Ctrl + X`.

3. Create a swap file.

To create a swap file of 1GB use this command:

```
sudo dd if=/dev/zero of=/swapfile bs=1024 count=1048576
```

Where:

- `/swapfile` is the path and name of the swap file. You can change this to something else.
- the number after `count` (1048576) equals 1GB. Increase it if you want to use a larger swap file. For example, multiply this number by 5 if you want to use a 5GB swap file (so use 5242880 as the `count=` value for a 5GB swap file).

If you use a different swap file name and path, make sure to use that instead of `/swapfile` in all the instructions that follow below.

4. Set the swap file permission to 600.

Use this so other users won't be able to read your swap file, which may contain sensitive information.

To set the swap file permission to 600, use this command:

```
sudo chmod 600 /swapfile
```

5. Format the newly created file as swap:

```
sudo mkswap /swapfile
```

6. Enable the newly created swap file:

```
sudo swapon /swapfile
```

To verify if the new swap file is in use, run:

```
swapon -s
```

It should output something like this:

```
Filename    Type   Size      Used   Priority
/swapfile   file   5242876   0      -2
```

7. Add the newly created swap file to `/etc/fstab`.

To use the new swap file each time you boot, you'll need to add it to the `/etc/fstab` file. Open `/etc/fstab` with a text editor (as root) like Nano:

```
sudo nano /etc/fstab
```

And in this file add the following line:

```
/swapfile none swap sw 0 0
```

To save the file (if you've used Nano command line editor) press Ctrl + O, then Enter. To exit Nano editor after you've saved the file press Ctrl + X. Again, remember to not modify anything else in the `/etc/fstab` file! Changing anything else in this file may prevent your system from booting!

8. This step is required for Ubuntu and Debian-based Linux distributions (I'm not sure if others need this too).

You need to edit the `/etc/initramfs-tools/conf.d/resume` file and comment out (add a `#` at the beginning of the line) the `RESUME=UUID=...` line. In my case, not doing this resulted in about 15-20 seconds of extra boot time. The `systemd-analyze blame` command didn't give any info as to why that's happening so I had to dig quite a bit to find out this is what's causing the boot delay.

Luckily I noticed a "*Gave up waiting for suspend/resume device*" message being displayed for a very brief moment while booting, which can be caused by not having the correct swap UUID in `/etc/initramfs-tools/conf.d/resume`.

This file is used when resuming from hibernation, and it caused boot delays because

we no longer have a swap partition.

To comment out this line in `/etc/initramfs-tools/conf.d/resume`, all you have to do is run the command below:

```
sudo sed -i 's/^RESUME=UUID/#RESUME=UUID/g' /etc/initramfs-tools/conf.d/resume
```

You'll also need to update initramfs and after that you're done:

```
sudo update-initramfs -u
```