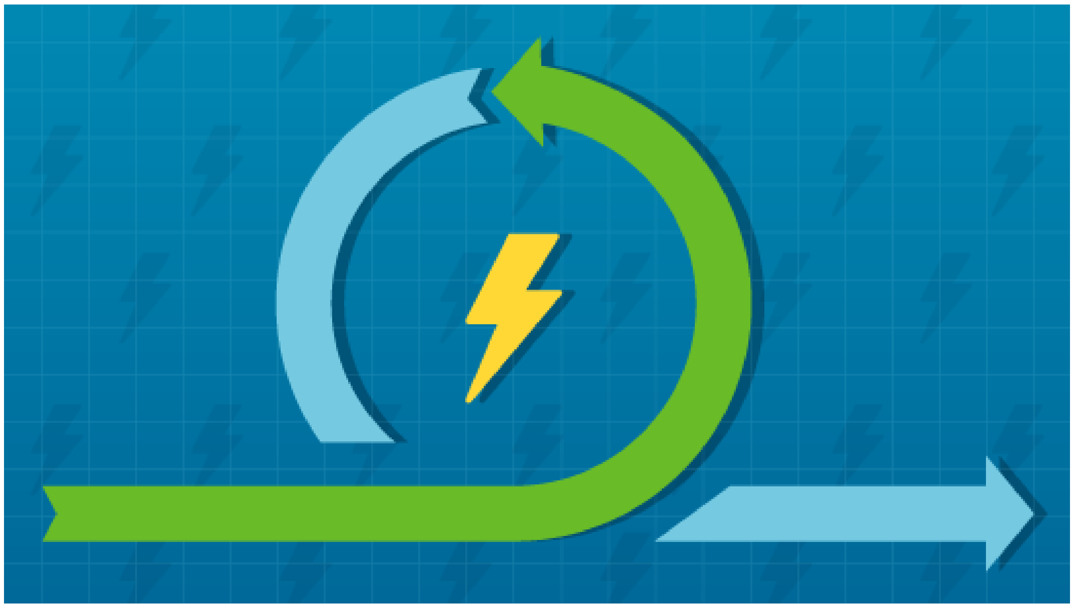
Swap space on Linux systems: A primer

opensource.com/article/18/9/swap-space-linux-systems

26 Sep 2018 David Both (Correspondent) Feed 170 up 5
comments



Swap space is a common aspect of computing today, regardless of operating system. Linux uses swap space to increase the amount of virtual memory available to a host. It can use one or more dedicated swap partitions or a swap file on a regular filesystem or logical volume.

There are two basic types of memory in a typical computer. The first type, random access memory (RAM), is used to store data and programs while they are being actively used by the computer. Programs and data cannot be used by the computer unless they are stored in RAM. RAM is volatile memory; that is, the data stored in RAM is lost if the computer is turned off.
Hard drives are magnetic media used for long-term storage of data and programs. Magnetic media is nonvolatile; the data stored on a disk remains even when power is removed from the computer. The CPU (central processing unit) cannot directly access the programs and data on the hard drive; it must be copied into RAM first, and that is where the CPU can access its programming instructions and the data to be operated on by those instructions. During the boot process, a computer copies specific operating system programs, such as the kernel and init or systemd, and data from the hard drive into RAM, where it is accessed directly by the computer's processor, the CPU.

Swap space

Swap space is the second type of memory in modern Linux systems. The primary function of swap space is to substitute disk space for RAM memory when real RAM fills up and more space is needed.

For example, assume you have a computer system with 8GB of RAM. If you start up programs that don’t fill that RAM, everything is fine and no swapping is required. But suppose the spreadsheet you are working on grows when you add more rows, and that, plus everything else that's running, now fills all of RAM. Without swap space available, you would have to stop working on the spreadsheet until you could free up some of your limited RAM by closing down some other programs.

The kernel uses a memory management program that detects blocks, aka pages, of memory in which the contents have not been used recently. The memory management program swaps enough of these relatively infrequently used pages of memory out to a special partition on the hard drive specifically designated for “paging,” or swapping. This frees up RAM and makes room for more data to be entered into your spreadsheet. Those pages of memory swapped out to the hard drive are tracked by the kernel’s memory management code and can be paged back into RAM if they are needed.

The total amount of memory in a Linux computer is the RAM plus swap space and is referred to as *virtual memory*.

Types of Linux swap

Linux provides for two types of swap space. By default, most Linux installations create a swap partition, but it is also possible to use a specially configured file as a swap file. A swap partition is just what its name implies—a standard disk partition that is designated as swap space by the `mkswap` command.

A swap file can be used if there is no free disk space in which to create a new swap partition or space in a volume group where a logical volume can be created for swap space. This is just a regular file that is created and preallocated to a specified size. Then the `mkswap` command is run to configure it as swap space. I don’t recommend using a file for swap space unless absolutely necessary.

Thrashing

Thrashing can occur when total virtual memory, both RAM and swap space, become nearly full. The system spends so much time paging blocks of memory between swap space and RAM and back that little time is left for real work. The typical symptoms of this are obvious: The system becomes slow or completely unresponsive, and the hard drive activity light is on almost constantly.

If you can manage to issue a command like `free` that shows CPU load and memory usage, you will see that the CPU load is very high, perhaps as much as 30 to 40 times the number of CPU cores in the system. Another symptom is that both RAM and swap space are almost completely allocated.

After the fact, looking at SAR (system activity report) data can also show these symptoms. I install SAR on every system I work on and use it for post-repair forensic analysis.

What is the right amount of swap space?

Many years ago, the rule of thumb for the amount of swap space that should be allocated on the hard drive was 2X the amount of RAM installed in the computer (of course, that was when most computers' RAM was measured in KB or MB). So if a computer had 64KB of RAM, a swap partition of 128KB would be an optimum size. This rule took into account the facts that RAM sizes were typically quite small at that time and that allocating more than 2X RAM for swap space did not improve performance. With more than twice RAM for swap, most systems spent more time thrashing than actually performing useful work.

RAM has become an inexpensive commodity and most computers these days have amounts of RAM that extend into tens of gigabytes. Most of my newer computers have at least 8GB of RAM, one has 32GB, and my main workstation has 64GB. My older computers have from 4 to 8 GB of RAM.

When dealing with computers having huge amounts of RAM, the limiting performance factor for swap space is far lower than the 2X multiplier. The Fedora 28 online Installation Guide, which can be found online at Fedora Installation Guide, defines current thinking about swap space allocation. I have included below some discussion and the table of recommendations from that document.

The following table provides the recommended size of a swap partition depending on the amount of RAM in your system and whether you want sufficient memory for your system to hibernate. The recommended swap partition size is established automatically during installation. To allow for hibernation, however, you will need to edit the swap space in the custom partitioning stage.

Table 1: Recommended system swap space in Fedora 28 documentation

Amount of system RAM	**Recommended swap space**	**Recommended swap with hibernation**

Amount of system RAM	**Recommended swap space**	**Recommended swap with hibernation**
less than 2 GB	2 times the amount of RAM	3 times the amount of RAM
2 GB - 8 GB	Equal to the amount of RAM	2 times the amount of RAM
8 GB - 64 GB	0.5 times the amount of RAM	1.5 times the amount of RAM
more than 64 GB	workload dependent	hibernation not recommended

At the border between each range listed above (for example, a system with 2 GB, 8 GB, or 64 GB of system RAM), use discretion with regard to chosen swap space and hibernation support. If your system resources allow for it, increasing the swap space may lead to better performance.

Of course, most Linux administrators have their own ideas about the appropriate amount of swap space—as well as pretty much everything else. Table 2, below, contains my recommendations based on my personal experiences in multiple environments. These may not work for you, but as with Table 1, they may help you get started.

Table 2: Recommended system swap space per the author

Amount of RAM	**Recommended swap space**
≤ 2GB	2X RAM
2GB – 8GB	= RAM
>8GB	8GB

One consideration in both tables is that as the amount of RAM increases, beyond a certain point adding more swap space simply leads to thrashing well before the swap space even comes close to being filled. If you have too little virtual memory while following these recommendations, you should add more RAM, if possible, rather than more swap space. As with all recommendations that affect system performance, use what works best for your specific environment. This will take time and effort to experiment and make changes based on the conditions in your Linux environment.

Adding more swap space to a non-LVM disk environment

Due to changing requirements for swap space on hosts with Linux already installed, it may become necessary to modify the amount of swap space defined for the system. This procedure can be used for any general case where the amount of swap space needs to be increased. It assumes sufficient available disk space is available. This procedure also assumes that the disks are partitioned in “raw” EXT4 and swap partitions and do not use logical volume management (LVM).

The basic steps to take are simple:

1. Turn off the existing swap space.
2. Create a new swap partition of the desired size.
3. Reread the partition table.
4. Configure the partition as swap space.
5. Add the new partition/etc/fstab.
6. Turn on swap.

A reboot should not be necessary.

For safety's sake, before turning off swap, at the very least you should ensure that no applications are running and that no swap space is in use. The `free` or `top` commands can tell you whether swap space is in use. To be even safer, you could revert to run level 1 or single-user mode.

Turn off the swap partition with the command which turns off all swap space:

```
swapoff -a
```

Now display the existing partitions on the hard drive.

```
fdisk -l
```

This displays the current partition tables on each drive. Identify the current swap partition by number.

Start `fdisk` in interactive mode with the command:

```
fdisk /dev/<device name>
```

For example:

```
fdisk /dev/sda
```

At this point, `fdisk` is now interactive and will operate only on the specified disk drive.

Use the fdisk `p` sub-command to verify that there is enough free space on the disk to create the new swap partition. The space on the hard drive is shown in terms of 512-byte blocks and starting and ending cylinder numbers, so you may have to do some math to determine the available space between and at the end of allocated partitions.

Use the `n` sub-command to create a new swap partition. fdisk will ask you the starting cylinder. By default, it chooses the lowest-numbered available cylinder. If you wish to change that, type in the number of the starting cylinder.

The `fdisk` command now allows you to enter the size of the partitions in a number of formats, including the last cylinder number or the size in bytes, KB or MB. Type in 4000M, which will give about 4GB of space on the new partition (for example), and press Enter.

Use the `p` sub-command to verify that the partition was created as you specified it. Note that the partition will probably not be exactly what you specified unless you used the ending cylinder number. The `fdisk` command can only allocate disk space in increments on whole cylinders, so your partition may be a little smaller or larger than you specified. If the partition is not what you want, you can delete it and create it again.

Now it is necessary to specify that the new partition is to be a swap partition. The sub-command `t` allows you to specify the type of partition. So enter `t`, specify the partition number, and when it asks for the hex code partition type, type 82, which is the Linux swap partition type, and press Enter.

When you are satisfied with the partition you have created, use the `w` sub-command to write the new partition table to the disk. The `fdisk` program will exit and return you to the command prompt after it completes writing the revised partition table. You will probably receive the following message as `fdisk` completes writing the new partition table:

```
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 16: Device or resource busy.

The kernel still uses the old table.

The new table will be used at the next reboot.

Syncing disks.
```

At this point, you use the `partprobe` command to force the kernel to re-read the partition table so that it is not necessary to perform a reboot.

```
partprobe
```

Now use the command `fdisk -l` to list the partitions and the new swap partition should be among those listed. Be sure that the new partition type is “Linux swap”.

It will be necessary to modify the /etc/fstab file to point to the new swap partition. The existing line may look like this:

```
LABEL=SWAP-sdaX   swap        swap    defaults        0 0
```

where `X` is the partition number. Add a new line that looks similar this, depending upon the location of your new swap partition:

```
/dev/sdaY         swap        swap    defaults        0 0
```

Be sure to use the correct partition number. Now you can perform the final step in creating the swap partition. Use the `mkswap` command to define the partition as a swap partition.

```
mkswap /dev/sdaY
```

The final step is to turn swap on using the command:

```
swapon -a
```

Your new swap partition is now online along with the previously existing swap partition. You can use the `free` or `top` commands to verify this.

Adding swap to an LVM disk environment

If your disk setup uses LVM, changing swap space will be fairly easy. Again, this assumes that space is available in the volume group in which the current swap volume is located. By default, the installation procedures for Fedora Linux in an LVM environment create the swap partition as a logical volume. This makes it easy because you can simply increase the size of the swap volume.

Here are the steps required to increase the amount of swap space in an LVM environment:

1. Turn off all swap.
2. Increase the size of the logical volume designated for swap.
3. Configure the resized volume as swap space.
4. Turn on swap.

First, let’s verify that swap exists and is a logical volume using the `lvs` command (list logical volume).

```
[root@studentvm1 ~]# lvs

  LV     VG                Attr       LSize  Pool   Origin Data%  Meta%  Move Log Cpy%Sync Convert

  home   fedora_studentvm1 -wi-ao----  2.00g

  pool00 fedora_studentvm1 twi-aotz--  2.00g               8.17   2.93

  root   fedora_studentvm1 Vwi-aotz--  2.00g pool00        8.17

  swap   fedora_studentvm1 -wi-ao----  8.00g

  tmp    fedora_studentvm1 -wi-ao----  5.00g

  usr    fedora_studentvm1 -wi-ao---- 15.00g

  var    fedora_studentvm1 -wi-ao---- 10.00g

[root@studentvm1 ~]#
```

You can see that the current swap size is 8GB. In this case, we want to add 2GB to this swap volume. First, stop existing swap. You may have to terminate running programs if swap space is in use.

```
swapoff -a
```

Now increase the size of the logical volume.

```
[root@studentvm1 ~]# lvextend -L +2G /dev/mapper/fedora_studentvm1-swap

  Size of logical volume fedora_studentvm1/swap changed from 8.00 GiB (2048 extents) to 10.00 GiB (2560 extents).

  Logical volume fedora_studentvm1/swap successfully resized.

[root@studentvm1 ~]#
```

Run the `mkswap` command to make this entire 10GB partition into swap space.

```
[root@studentvm1 ~]# mkswap /dev/mapper/fedora_studentvm1-swap

mkswap: /dev/mapper/fedora_studentvm1-swap: warning: wiping old swap signature.

Setting up swapspace version 1, size = 10 GiB (10737414144 bytes)

no label, UUID=3cc2bee0-e746-4b66-aa2d-1ea15ef1574a

[root@studentvm1 ~]#
```

Turn swap back on.

```
[root@studentvm1 ~]# swapon -a

[root@studentvm1 ~]#
```

Now verify the new swap space is present with the list block devices command. Again, a reboot is not required.

```
[root@studentvm1 ~]# lsblk

NAME                                    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT

sda                                       8:0    0   60G  0 disk

|-sda1                                    8:1    0    1G  0 part /boot

`-sda2                                    8:2    0   59G  0 part

  |-fedora_studentvm1-pool00_tmeta   253:0    0    4M  0 lvm

  | `-fedora_studentvm1-pool00-tpool 253:2    0    2G  0 lvm

  |   |-fedora_studentvm1-root       253:3    0    2G  0 lvm  /

  |   `-fedora_studentvm1-pool00     253:6    0    2G  0 lvm

  |-fedora_studentvm1-pool00_tdata   253:1    0    2G  0 lvm

  | `-fedora_studentvm1-pool00-tpool 253:2    0    2G  0 lvm

  |   |-fedora_studentvm1-root       253:3    0    2G  0 lvm  /

  |   `-fedora_studentvm1-pool00     253:6    0    2G  0 lvm

  |-fedora_studentvm1-swap           253:4    0   10G  0 lvm  [SWAP]

  |-fedora_studentvm1-usr            253:5    0   15G  0 lvm  /usr

  |-fedora_studentvm1-home           253:7    0    2G  0 lvm  /home

  |-fedora_studentvm1-var            253:8    0   10G  0 lvm  /var

  `-fedora_studentvm1-tmp            253:9    0    5G  0 lvm  /tmp

sr0                                      11:0    1 1024M  0 rom

[root@studentvm1 ~]#
```

You can also use the `swapon -s` command, or `top`, `free`, or any of several other commands to verify this.

```
[root@studentvm1 ~]# free

              total        used        free      shared  buff/cache   available

Mem:        4038808      382404     2754072        4152      902332     3404184

Swap:      10485756           0    10485756

[root@studentvm1 ~]#
```

Note that the different commands display or require as input the device special file in different forms. There are a number of ways in which specific devices are accessed in the /dev directory. My article, Managing Devices in Linux, includes more information about the /dev directory and its contents.