How to Add a Swap File on an Arch Linux Cloud Server

Authored by: ***ASPHostServer Administrator*** *[asphostserver@gmail.com]*
Saved From: http://faq.asphosthelpdesk.com/article.php?id=206

Introduction

An easy way to increase the responsiveness of your cloud server is to add some swap space. Swap is an area of the hard drive that can be used to temporarily hold information that is usually stored in RAM. The result is that more RAM is available for your currently running processes.

Under the Hood

The Linux kernel stores information in RAM in small chunks known as "pagesâ€•. A program can have a large number of pages taking up space in RAM. When a new process needs access to the physical memory, new pages are created as necessary. But what happens when there is not any more space in RAM for more pages?

If a program hasn"t used all of the pages it has been allocated in RAM, the unused pages can be removed and the space can be reallocated. This is because those page files can be easily recreated if they are needed. For example, if program A has been allocated 10 pages, but has only written to 7, then 3 can be destroyed temporarily and recreated when they are actually needed. In the mean time, the space that those 3 pages were using can be used by other programs.

So, where does swap come in?

Sometimes the system must remove pages from RAM containing data that can"t be recreated at will. The information written to them isn"t readily available on the hard drive. When the system removes pages in this state, it needs a place to put them so that it doesn"t lose that information. That place is called "swapâ€•.

There is nothing very complex about a swap file or a swap partition. It is simply an area that is devoted to temporarily storing data from RAM. This can be state information from applications, cached data, or anything else that can occupy RAM. The operating system will literally swap information from RAM to the swap space and vice versa whenever necessary.

Check for Swap Space

We will need to be logged in as root to complete the steps in this tutorial.

```
su
```

Before we create our own swap file, we want to check to see if there"s already a swap space enabled on the system. We can easily do this in a number of ways. Let"s try the "swaponâ€• command first.

```
swapon -s
```

```
swapon: stat failed -s: No such file or directory
```

As you can see, on a clean Arch Linux server, this command doesn"t produce any output. The command has executed correctly, but it hasn"t found any swap space.

Let''s try a different way of checking swap that will provide us with more visual feedback.

```
free -m

             total       used       free     shared    buffers     cached
Mem:           494         60        434          0          6         28
-/+ buffers/cache:         25        469
Swap:            0          0          0
```

As expected, swap is at 0 in the total column. This confirms what the "swaponâ€• command has told us.

Check the File System

There are many different opinions about the ideal size of a swap area. Most often it is adequate to either match or double the amount of RAM you have (unless you have an extremely large amount of RAM, in which case you can allocate less swap space). In the end, it comes down to personal preference and the decision is usually influenced by the amount of available RAM and hard drive space on your machine.

For this guide, we have 512MB of RAM and a 20GB hard drive. If we were to set up a swap space to be twice as large as our RAM, it would be a significant portion of our hard drive. So instead, we are going to match the amount of RAM we have and create a 512MB swap space.

Before we can actually create our swap space, we need to check that we have enough space on our hard drive available. Let''s check that now with the "dfâ€• command.

```
df -h

Filesystem      Size  Used Avail Use% Mounted on
/dev/vda         20G  965M   18G   6% /
dev             240M     0  240M   0% /dev
run             248M  212K  248M   1% /run
tmpfs           248M     0  248M   0% /dev/shm
tmpfs           248M     0  248M   0% /sys/fs/cgroup
tmpfs           248M     0  248M   0% /tmp
```

If we look at the first line, we can see our 20GB hard drive and roughly 18GB of that is unused. That is plenty of space to create our 512MB swap space.

Create and Enable the Swap File

We are now ready to create our swap space. We will discuss two different ways to accomplish this. First, we''ll use the "ddâ€• command.

```
dd if=/dev/zero of=/swapfile bs=1M count=512

512+0 records in
512+0 records out
```

```
536870912 bytes (537 MB) copied, 1.96268 s, 274 MB/s
```

Let"s confirm that the file was created correctly.

```
ls -lh /swapfile
```

```
-rw-r--r-- 1 root root 512M Jun 28 18:41 /swapfile
```

As you can see, we now have a 512MB file called /swapfile. Let"s try creating the swap file a different way. We"ll remove the swap file we just created so we can try again.

```
rm /swapfile
```

The "fallocateâ€• program can create swap files faster than "ddâ€•. As an added perk, its syntax is also easier to remember.

```
fallocate -l 512M /swapfile
```

Again, we can see that there is now a swap file of the correct size.

```
ls -lh /swapfile
```

```
-rw-r--r-- 1 root root 512M Jun 28 18:54 /swapfile
```

We now have a file in the root partition of our hard drive called "swapfileâ€•, but we can see that the system isn"t recognizing or using it yet.

```
free -m
```

```
             total       used       free     shared    buffers     cached
Mem:           494         41        453          0          1         13
-/+ buffers/cache:         25        468
Swap:            0          0          0
```

Right now, the file is just a regular file. We may have called it "swapfileâ€•, but Linux does not know to use it as swap yet. We have to format it as swap and enable it.

Before we do that though, we need to complete a very important step. We need to change the permissions of the file so that only root can read and write to the file. A swap file that can be read or written to by non-root users is a very large security vulnerability. As you saw when we ran "lsâ€•, it is currently readable by anyone on the system. Let"s change that now.

```
chmod 600 /swapfile
```

If we re-check the file, we can see that the permissions are now correct.

```
ls -lh /swapfile
```

```
-rw------- 1 root root 512M Jun 28 18:54 /swapfile
```

Now that our file is secure, we can format it.

```
mkswap /swapfile
```

```
Setting up swapspace version 1, size = 524284 KiB
no label, UUID=61f42c98-d1ef-4f27-84c2-713c5f63d434
```

Finally, we can enable the file so that the system knows to begin using it.

```
swapon /swapfile
```

Let''s check to see that our file is recognized and being used.

```
free -m
```

```
             total       used       free     shared    buffers     cached
Mem:           494         42        452          0          1         14
-/+ buffers/cache:         25        468
Swap:          511          0        511
```

Great! Our swap file is recognized and the operating system will begin using it to store data from RAM as necessary.

We aren''t done yet though. The operating system needs to know that it is safe to use this file for swap every time it boots up. We can accomplish that by editing the ''/etc/fstabâ€• file.

```
nano /etc/fstab
```

```
# Paste this information at the bottom of the file
/swapfile none swap defaults 0 0
```

Tweaking Swap Settings

There are a number of things to keep in mind when using swap. Because swap space is on a hard drive, it is much slower than RAM. That means that you only want the operating system to use it when it needs to. If pages are being moved to your swap space well before you are out of RAM, you will experience a substantial slowdown in your operations.

The parameter that controls how often a system moves items out of RAM and into a swap space is known as ''swappinessâ€•. Its value can be anything between 0 and 100. With values close to zero, the kernel will choose not to swap unless absolutely necessary. With values approaching 100, the system will swap out files much earlier. This can lead to speed issues because the information must be fetched from a hard drive instead of RAM.

We can find out what our current swappiness setting by entering the following command:

```
cat /proc/sys/vm/swappiness
```

```
60
```

Our system defaults to a swappiness value of 60. This is a good default for a server. If we were adjusting the swappiness of a desktop, we would probably want to move it closer to 0. We can try different swappiness values using the "sysctlâ€• command.

```
sysctl vm.swappiness=10
```

```
vm.swappiness = 10
```

We can see that the system has accepted our new value for the swappiness parameter.

```
cat /proc/sys/vm/swappiness
```

```
10
```

If we want to default this setting, we can edit the "/etc/sysctl.confâ€• file.

```
nano /etc/sysctl.conf
```

```
# Search for the vm.swappiness setting.  Uncomment and change it as necessary.
vm.swappiness=10
```

This will allow our settings to persist through reboots. Feel free to play with this parameter to achieve optimal swapping for your situation.

Another related parameter that we might want to adjust is the "vm.vfs_cache_pressureâ€• setting. Altering the cache pressure value will change the system"s preference for keeping inode and dentry cached information over other kinds of information. This means that the operating system will retain information about the file system longer, which happens to be very costly to look up and very frequently requested. This will usually lead to a performance gain.

Again, we can check the current value by looking into the /proc directory.

```
cat /proc/sys/vm/vfs_cache_pressure
```

```
100
```

The current settings move inode information out of cache more quickly than we want it to. A cache pressure of 50 would be much better for our purposes. We can change these settings the same way as we did with swappiness.

```
sysctl vm.vfs_cache_pressure=50
```

```
vm.vfs_cache_pressure = 50
```

Again, we can make this persist through reboots by adding it to the "/etc/sysctl.confâ€• file.

```
nano /etc/sysctl.conf
```

```
# Search for the vm.vfs_cache_pressure setting.  Uncomment and change it as
necessary.
vm.vfs_cache_pressure = 50
```

This makes our cache pressure configuration permanent.