Accessing Those Old macOS Volumes

How to mount and access the storage drive of an old Mac via Linux.

By Petros Koutoupis

Nowadays, all newly installed versions of the Macintosh OS pre-format the local storage drive with the Apple File System (APFS). Before APFS, there was the Hierarchical File System Plus (HFS+). For quite a long time (since at least 1998), this has been the default filesystem for all that was and continues to be Macintosh. If you are like many others transitioning from older Macintosh devices and looking to move toward a Linux-based one, you may find yourself circling back to that old set of storage volumes containing many years worth of data. Fortunately, the Linux environment contains such tools to be able to accomplish this.

The Tools for the Job

In this scenario, let's say this storage device, be it a Hard Disk Drive (HDD) or Solid State Drive (SSD) is connected either externally or installed internally as a secondary device to your new and current Linux platform. You'll first need to install a base set of packages. On Debian or Ubuntu, it would look something like this:

```
$ sudo apt install hfsplus hfsprogs
```

Seeing how HFS+ is a very dated and very feature-limited filesystem, I doubt anyone will be formatting a new volume with that filesystem. But let's say you're using an external volume that needs to hop from Linux to a Mac and back. You can format it with the following `mkfs` command:

```
$ sudo /sbin/mkfs.hfs /dev/sdb1
Initialized /dev/sdb1 as a 131072 MB HFS Plus volume
```

To mount it:

```
$ sudo mount –t hfsplus /dev/sdb1 /mnt
```

But, what if your volume was the main operating system storage drive of that old Macintosh or MacBook? How do you mount it and access it via Linux?

In the following example, a dump of the partition layout on the macOS reveals that the main operating system partition is set to disk0s2:

```
$ diskutil list
/dev/disk0 (internal):
   #:                       TYPE NAME                    SIZE       IDENTIFIER
   0:      GUID_partition_scheme                        121.3 GB   disk0
   1:                        EFI EFI                     314.6 MB   disk0s1
   2:                  Apple_HFS Macintosh HD           96.5 GB    disk1s2
   3:                  Apple_Boot Recovery              522.7 MB   disk1s3
```

Moving that same volume to Linux, the “Macintosh HD” label would read “Apple Core Storage” (visible via an `fdisk` or `parted` partition table dump), and assuming that the device identifies as /dev/sdb when connected, the above partition will map to /dev/sdb2.

You should be able to mount the filesystem with read-only access using the above `mount` command, but if you need to enable write access, you can do it in one of two ways.

The Brute-Force Method

Mount the HFS+ drive with the following command:

```
$ sudo mount –t hfsplus –o force,rw /dev/sdb2 /mnt
```

Or remount:

```
$ sudo mount –t hfsplus –o remount,force,rw /mnt
```

This approach is not necessarily considered safe, and it's strongly advised to run a filesystem check quite regularly either before mounting the volume or after unmounting it:

```
$ sudo fsck.hfsplus –f /dev/sdb2
```

Disabling the Journal

Now, this too is not a recommended approach, as filesystem journaling plays a very important role in the filesystem and is intended to improve filesystem reliability (and data consistency), but with this method, to mount the HFS+ filesystem in a Linux environment successfully, you'll need to do just this. First, in macOS, identify both the device and the partition. If you were to use the same device as in the example above, the command would look something like this:

```
$ sudo diskutil disableJournal disk0s2
```

Note: there is a known issue with some versions of OS X that doesn't properly register the `disableJournal` *command unless you run the* `enableJournal` *command before it.*

After disabling the journal from a macOS environment, take the drive to a Linux environment and identify its partition.

Note: again, disabling the journal is not considered safe, and it is strongly advised to run filesystem checks quite regularly either before mounting the volume or after unmounting it.

Home Directory Shenanigans

For the purpose of this article, let's use the same partition as above (/dev/sdb2). Mount the device. If this once hosted your macOS home directory, you'll immediately notice that you're unable to read or write from/to the volume at that

destination, unless you are running as `root`. You'll be able to enable read/write access to this home directory by changing your User ID (UID) to match the UID used by your user under the macOS environment. The cleanest way to do this is by creating a new (and maybe temporary) user in your Linux environment.

Typically, the UID used by the *first* user under macOS is 501. If you still have the storage device connected to your Macintosh, open a terminal and type `id`. The user UID will be displayed:

```
$ id
uid=501(petros) gid=20(staff) groups=20(staff),12(everyone),
↪61(localaccounts),79(_appserverusr),80(admin),
↪81(_appserveradm),98(_lpadmin),701(com.apple.sharepoint
↪.group.1),33(_appstore),100(_lpoperator),204
↪(_developer),250(_analyticsusers),395(com.apple.
↪access_ftp),398(com.apple.access_screensharing),
↪399(com.apple.access_ssh)
```

Remember this number.

Create the new (or temporary) user:

```
$ sudo useradd -d /home/osxuser -m -s /bin/bash -G
 ↪adm,sudo osxuser
```

Create the password for this new user:

```
$ sudo passwd osxuser
```

Log in as the new user:

```
$ su osxuser
```

Change your Linux user's UID to 501:

```
$ sudo usermod ––uid 501 osxuser
```

And, fix your home directory permission to reflect this change:

```
$ sudo chown –R 501:osxuser /home/osxuser
```

Now you should be able to read/write to both your Mac and Linux user's home directory, regardless of the operating system you are using and logged in to. ■

Petros Koutoupis, *LJ* Editor at Large, is currently a senior performance software engineer at Cray for its Lustre High Performance File System division. He is also the creator and maintainer of the RapidDisk Project. Petros has worked in the data storage industry for well over a decade and has helped pioneer the many technologies unleashed in the wild today.

Send comments or feedback
via http://www.linuxjournal.com/contact
or email ljeditor@linuxjournal.com.