Create custom FreeBSD .img from original FreeBSD-13.0-RELEASE-amd64-memstick.img

fumanchu

-
- #1

Hello,

I am looking at taking the original FreeBSD-13.0-RELEASE-amd64-memstick.img image file, mounting it, adding a few new .sh scripts, and then creating a new bootable .img file. I am changing nothing else...

I have been able to mount the original .img using the following commands...

Bash:

```
$ mdconfig -a -t vnode -f /path/to/FreeBSD-11.0-RELEASE-amd64-mini-memstick.img -o readonly -u 0
$ mount -r /dev/md0s2a /mnt
```

I have then created a directory and added my .sh scripts...

And then I have been able to unmount using the following commands...

Bash:

```
$ umount /mnt
$ mdconfig -d -u 0
```

I am stuck at how to create the new bootable .img file...

Thoughts?

-
- #2

You don't need to create a new bootable

```
.img
```

file. The one you have added scripts to is good for use.

-
- #3

It is even possible to grow the image by dd'ing zeros to the image file and then mount your image via memory disk and use tools to grow the partition and filesystem mounted in your memory disk.. UFS has a good amount of padding so you can add some files to the stock image.

-
- #4

While it is correct what T-Daemon and Phishfry said, you need to omit the readonly flags on your respective commands, since otherwise, I doubt that anything becomes written to the image.

```
mdconfig -a -u 0 -t vnode -f /path/to/FreeBSD-11.0-RELEASE-amd64-mini-memstick.img
mount /dev/md0s2a /mnt
```

-
- #5

I should mention if you expand an image (with zeros) that uses a GPT filesystem scheme you will need to run gpart recover.
GPT puts a boot record at begining of disk and at the end. When you expand a GPT image you lose the GPT in the last sectors. So `gpart recover ${DISK_NAME}` will fix it when mounted as Memory Disk.

- Thread Starter
-
- #6

Thanks for the great information... I am really new at all this so please bear with me...

As T_Daemon mentioned, I would not have to create a new .img as I can simply use the one I already have. I am a bit confused on that statement... I assume that once I unmount /mnt, the original .img is automatically updated with my changes? If that is true, that is pretty cool...

Then as Phishfry mentioned, it is possible to grow image... I assume this is only needed if I add enough new stuff to the original .img file that it's original size would be exceeded? If I simple added a couple small .sh scripts, the original .img size would most likely not be exceeded; where as, if I added a full blown package, say something like vim, the new size of the .img would be exceeded so I would have to expand the size? If this is true, I will need to dig into how to expand the .img as Phishyfry also mentioned.

-
- #7

> I assume that once I unmount /mnt, the original .img is automatically updated with my changes? If that is true, that is pretty cool...

Yes. It's an image of a disk, mdconfig(8) makes it so you can access it as if it was a 'real' disk.

> Then as Phishfry mentioned, it is possible to grow image... I assume this is only needed if I add enough new stuff to the original .img file that it's original size would be exceeded?

Yes, the image is relatively small. So you could run out of disk space on it.

-
- #8

Think of the IMG file as an container. You can grow this container and adjust the filesystems to suit.

> I assume this is only needed if I add enough new stuff to the original .img file that it's original size would be exceeded?

Yes that is true.

Use `df -h` to see how full the container gets.
UFS has around 7% padding so you could theoretically see up to 107% disk usage.
In reality you never want to get it that full. Maybe 105%. Very ill consequences if you meet the real end of disk.

- Thread Starter
-
- #9

Many thanks for the great information! I will give this all a try.

- Thread Starter
-
- #10

So far so good as I am to the part where I now need to expand the memory disk to the new size as the files that I need to add definitely cause an out of disk space error...

Is there somewhere I can be pointed to for information on how to expand the memory disk? I have tried gpart's resize command; however, that does not seem to work... well I might not be using it correctly...

When I try gpart, I get this...

Bash:

```
$ mdconfig -a -t vnode -f $HOME/FreeBSD-13.0-RELEASE-amd64-memstick.img -u 0
$ gpart show /dev/md0
->       1  2130664  md0  MBR   (1.0G)
         1    66584    1  efi   (33M)
     66585  2064080    2  freebsd  [active]  (1.0G)
```

I assume that I need to expand the freebsd subpartition...

-
- #11

> Is there somewhere I can be pointed to for information on how to expand the memory disk? I have tried gpart's resize command; however, that does not seem to work...

You need first to increase the size of the image before resizing it with gpart(8). See truncate(1).

> I assume that I need to expand the freebsd subpartition...

You need to resize (expand) two times.

The partition scheme and partition structure of the installer image (FreeBSD calls MBR partitions slices, slices can have partitions):

Code:

```
MBR                                         md0
slice 1       efi                           md0s1
slice 2       freebsd                       md0s2
        BSD partition scheme                md0s2
        slice 2 partition 1  freebsd-ufs    md0s2a
```

This is the order of commands to increase the size of the FreeBSD installer image:

Code:

```
truncate -s +<size>m FreeBSD-...img  # m = Megabyte see truncate(1)

mdconfig  -u 0 FreeBSD-...img   # You can skip all other options, even '-u 0'.
                                # See 'mdconfig file' in mdconfig(8).

gpart resize -i 2 md0
gpart resize -i 1 md0s2

growfs md0s2a

mount /dev/md0s2a /mnt
Add files
umount /mnt

mdconfig -du 0
```

Image ready for use

For details see the manuals of the commands.

- Thread Starter
-
- #12

Many thanks! I will give that a try and report back...

-
- #13

Truncate should work but in the past I have just dd'ed zeros to the tail end of the file.

VM File sizing

I am new to VM's so I wonder what is the best strategy for VM file sizing. For example I am building a Poudiere VM and I truncated my VM for 80GB. What if I need more later? Can I DD my VM image into another larger image and grow the filesystem on the VM side? So I can probably grow a VM but...

forums.freebsd.org

This shows VM image manipulation. Same concept.

- Thread Starter
-
- #14

Okay, reporting back my results.

I am able to add my files to the the .img; however, when I try and use this .img to install FreeBSD, I do not have my files installed.

What I have done:

1. I downloaded FreeBSD-13.0-RELEASE-amd64-memstick.img.
2. I copied FreeBSD-13.0-RELEASE-amd64-memstick.img to FreeBSD-13.0-RELEASE-amd64-memstick-NEW.img
3. Installed **portupdate** package using
4. Used the **pkg_fetch** command to download the necessary packages including all their dependencies. By default the packages are downloaded to */usr/ports/packages*.
5. After downloading all the packages, I calculated the size.
6. I then used the **truncate** command to increase the size of the .img by that amount

   Bash:

   ```
   truncate -s +130m FreeBSD-13.0-RELEASE-amd64-memstick-NEW.img
   ```

7. I then created a memory disk from the .img

   Bash:

   ```
   mdconfig -u 0 FreeBSD-13.0-RELEASE-amd64-memstick-NEW.img
   ```

8. I then ran **gpart** on each of the partitions
9. I then grew the filesystem using the **growfs** command
10. Next, I mounted the memory disk using the mount command
11. Next, I went to the */mnt/us*r and created a new directory for the packages I had downloaded
12. Next, I copied the files from */usr/ports/packages* to */mnt/usr/packages*

   Bash:

   ```
   cp /usr/ports/packages /mnt/usr/packages
   ```

13. Next, I umounted the memory disk
14. Next, I removed the memory disk

After all this I created the memory disk from the NEW.img as well as mounted it to /mnt again just to make sure my files were then and sure enough, they were. However, when I attempt to install FreeBSD from this new .img, my /usr/packages directory and files does not exist on the machine...

Any idea?

-
- #15

> 4. Used the **pkg_fetch** command to download the necessary packages including all their dependencies. **By default the packages are downloaded to */usr/ports/packages*.**

Not related to the problem: By default pkg-fetch(8) places fetched packages in internal cache (

```
/var/cache/pkg
```

), unless '-o <destdir>' is given as option.

> However, when I attempt to install FreeBSD from this new .img, my /usr/packages directory and files does not exist on the machine

I can't reproduce missing

```
/usr/packages
```

directory and files on installation image. After executing all steps and dd(1)'ing the

```
.img
```

on a USB stick and mounting or booting it, the directory and files are at their places.

Are you sure you copied the

```
*-NEW.img
```

on installation media, not an original, unmodified image?

- Thread Starter
-
- #16

I will check again in the morning and report my findings. I am pretty sure I copied the -NEW.image but I could be mistaken.

-
- #17

> By default pkg-fetch(8) places fetched packages in internal cache (/var/cache/pkg), unless '-o <destdir>' is given as option.

And even if you specify a directory with "-o" the package files actually end up in -o <destdir/All>

- Thread Starter
-
- #18

Okay, here is what I did to confirm that my files are not transferred to a fresh install...

1. I copied the -NEW.image onto a flash-drive.
2. I took that flash-drive to another FreeBSD machine, created the memory disk, mount it, and looked at its contents.
3. All my custom files were indeed there!
4. I then burned that confirmed image onto a new flash-drive so I can boot from it.
5. For grins, I mounted the newly burned flash-drive and took a look at the contents to make sure my custom files were still there... They were!
6. I then booted another machine using that flash-drive and installed a fresh copy if FreeBSD...
7. After the install finished, I took a look at the new install to see if my custom files were there... They were not!

Not sure what I missed. Maybe the actual FreeBSD installer is not picking up my custom files for some reason... More investigation needed...

-
- #19

> Okay, here is what I did to confirm that my files are not transferred to a fresh install...
> ...
> 6. I then booted another machine using that flash-drive and installed a fresh copy if FreeBSD...
> 7. After the install finished, I took a look at the new install to see if my custom files were there... They were not!

Until now I was under the impression we are talking about the presents of files (packages) on the installation image, not about a automated director creation on the newly installed system and transfer of packages (custom files) there from the installation media.

> Maybe the actual FreeBSD installer is not picking up my custom files for some reason...

There is no [bsdinstall(8)]() menus installation procedure that creates automatic a package directory on the target system and initiates a transfer of custom files.

If you want that you need to do it manually or scripted (see SCRIPTING in [bsdinstall(8)]()).

- Thread Starter
-
- [#20]()

Hum...

Since our users may or may not have Internet access, they can simply take this modified bootable memstick image, burn it on a flash/thumb drive, stick in into a machine, boot from it, and go through the normal installation process. My plan was that the installer (or whatever it is called) would have simply copied my files to the destination drive since they are already inside the image alongside all the other files... I would then instruct the user, via documentation, on how to run the custom script. This script that they would run would install all the pre-downloaded packages as well as perform other tasks like, create a pre-defined user, harden the operating as needed, etc.

-
- [#21]()

> My plan was that the installer (or whatever it is called) would have simply copied my files to the destination drive since they are already inside the image alongside all the other files...

That's not how the installer ([bsdinstall(8)]()) works. The new systems base and kernel are not copied from the installer images file system but extracted from

```
/usr/freebsd-dist/base.txz
```

and

```
kernel.txz
```

.

> I would then instruct the user, via documentation, on how to run the custom script. This script that they would run would install all the pre-downloaded packages ...

Can't the script install pre-downloaded packages from

```
$INSTALLER_IMAGE/usr/packages
```

instead of

```
$NEW_SYSTEM/usr/packages
```

?

If you insist on having a

```
$NEW_SYSTEM/usr/packages/*txz
```

directory and files, unless nobody else has a better idea, you could manipulate

```
$INSTALLER_IMAGE/usr/freebsd-dist/base.txz
```

.

Follow steps bellow as root, assuming a memory disk is created from

```
*memstick.img
```

and file system mounted:

Code:

```
cd /tmp
mkdir a
mv /mnt/usr/freebsd-dist/base.txz  .

tar xf base.txz -C a
mkdir a/usr/packages
cp /usr/ports/packages/* a/usr/packages

cd a
tar acf /mnt/usr/freebsd-dist/base.txz  .

cd /mnt/usr/freebsd-dist
rm MANIFEST
sh /usr/src/release/scripts/make-manifest.sh  *.txz  >  MANIFEST

umount ...
mdconfig ...
```

- Thread Starter
-
- #22

-
- #23

That won't work, it won't create

```
$NEW_SYSTEM/usr/packages/*.txz
```

. If you want them there the only way to do that, as I see it, is to add the pre-downloaded packages in

```
base.txz
```

. When the new system is installed from there it will have the packages directory and files.

-
- #24

~~Ah, sorry, that works too. I recalled it otherwise.~~

What am I talking. It won't work. Sorry, I hadn't much sleep last night, someone close to me has passed away.

- Thread Starter
-
- #25

No problem. I totally understand the lack of sleep issue.

I am looking into your proposed solution with modifying *base.txz*...