Create a Custom RaspberryPi OS Image

Mathieu ABATI (mathieu-abati.com)

06 / 09 / 2025

Sommaire

Create a Custom RaspberryPi OS Image

In this tutorial, we will see a method to create a custom RaspberryPi OS image, in which we can pre-install our own packages and configurations.

The steps are detailed step-by-step, but for automation via a script, you can find an example here.

Preparing the Base Image

First, you need to download a RaspberryPi OS image from the official page.

Then, decompress it:

```
unxz -k -d raspios.img.xz -c > "raspios.img"
```

Preparing the Filesystem

It may be necessary to enlarge the root partition of the image to be able to install new packages. We start by adding zeros to the end of the image (here, 512 MB):

```
dd if=/dev/zero bs=1M count=512 >> "raspios.img"
```

and we resize the root partition accordingly:

```
sudo parted "raspios.img" resizepart 2 100%
```

Now, we need to modify the filesystem of this partition to occupy all the available space. We get the partition information:

```
parted -s "raspios.img" unit B print
```

we get something like:

```
Model:  (file)
Disk raspios.img: 2558525440B
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start       End          Size         Type     File system
    Flags
 1      8388608B    545259519B   536870912B   primary  fat32        lba
 2      545259520B  2558525439B  2013265920B  primary  ext4
```

The information we are interested in is the start of the root partition (ext4), so: `545259520B`. Below, be sure to remove the trailing `B`. We use this information to create a loop device:

```
sudo losetup --show -f -o 545259520 raspios.img
```

and note the path of the created device, here: `/dev/loop0`.

It is recommended to check the filesystem integrity:

```
sudo e2fsck -p -f /dev/loop0
```

We resize the filesystem to the size of the partition:

```
sudo resize2fs /dev/loop0
```

there should be a message confirming that the resizing was successful.

Mounting the Filesystems

We will now mount the boot partition and the root partition.

First, we need to create the loop device for the boot partition, just as we did for the root partition. We need to get the start of the partition given by the `parted` command we used earlier, in our case `8388608B`. We remove the trailing `B`, and create the loop device:

```
sudo losetup --show -f -o 8388608 raspios.img
```

and note the path of the created device, here: `/dev/loop1`.

We can now mount our partitions:

```
mkdir boot root
sudo mount /dev/loop1 boot
sudo mount /dev/loop0 root
sudo mkdir -p root/boot
sudo mount --bind boot root/boot
```

Preparing the chroot

We need to mount the special directories in the root filesystem:

```
sudo mount --bind /dev root/dev
sudo mount --bind /sys root/sys
sudo mount --bind /proc root/proc
```

also, we copy the DNS server information for domain name resolution:

```
sudo cp /etc/resolv.conf root/etc/resolv.conf
```

Modifying the Filesystem via chroot

We can now do:

```
sudo chroot root /bin/bash
```

From there, you can install packages in the image:

```
sudo apt update
sudo apt install ...
```

or modify the system configuration.

Finalizing the Image Creation

To exit the chroot, press `CTRL + D`. Then, remove the resources that are no longer needed and unmount the special filesystems:

```
sudo rm -f root/etc/resolv.conf
sudo umount root/dev
sudo umount root/sys
sudo umount root/proc
sudo umount root/boot
sudo umount root
sudo umount boot
```

Destroy the loop devices:

```
sudo losetup -d /dev/loop0
sudo losetup -d /dev/loop1
```

Finally, compress the image:

```
xz -k -c raspios.img > raspios_custom.img.xz
```