Creating Initramfs

Created By	Ⓓ Daniil A (Kiky Tokamuro)
Last Edited	@Sep 13, 2019
Tags	linux

Initramfs, short for “initial RAM file system”, is the receiver of initrd (initial ramdisk). This is the cpio (copy in and out) archive of the source file system, which is loaded into memory during the Linux startup process. Linux copies the contents of the archive to rootfs (which can be based on ramfs or tmpfs), and then runs init. Init is designed to perform certain tasks before the real or final file system is installed on top of rootfs. Thus, initramfs should contain all the device drivers and tools needed to install the final root file system.

1. Download busybox (you can download a newer version):

```
wget https://busybox.net/downloads/busybox-1.26.2.tar.bz2
tar -xvf busybox-1.26.2.tar.bz2
```

2. We collect busybox from source codes:

```
cd busybox-1.26.2
make defconfig
make menuconfig
```

In the Busybox Settings menu, select Build Options, and check the box next to Build BusyBox as a static binary (no shared libs). Next, specify the output folder for binaries and collect busybox:

```
make
make CONFIG_PREFIX=./../busybox_rootfs install
```

3. Create a directory hierarchy for initramfs:

```
mkdir -p initramfs/{bin,dev,etc,home,mnt,proc,sys,usr}
cd initramfs/dev
sudo mknod sda b 8 0
sudo mknod console c 5 1
```

Also, copy everything from the busybox_rootfs folder to the initramfs folder. Next, create an init file in the root of initramfs, and write the following into it:

```
#!/bin/sh
mount -t proc none /proc
mount -t sysfs none /sys
exec /bin/sh
```

And we give him the right to executing:

```
chmod +x init
```

4. Create initramfs itself:

```
find . -print0 | cpio --null -ov --format=newc > initramfs.cpio
gzip ./initramfs.cpio
```

5. Download and build the kernel (you can download a newer version):

```
wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.11.6.tar.xz
tar -xvf linux-4.11.6.tar.xz
```

```
make x86_64_defconfig
make kvmconfig
make -j2
```

The kernel image will be located in /arch/x86_64/boot/bzImage.

6. Next, copy somewhere our initramfs and the kernel, go into this directory and run qemu:

```
qemu-system-x86_64 -kernel ./bzImage -initrd
./initramfs.cpio.gz -nographic -append "console=ttyS0"
```

7. If everything was done correctly, the kernel will boot and the shell will start.