Android boot.img manipulation (//k.japko.eu/boot-img-manipulation.html)

Date Sun 16 February 2014 Tags android (//k.japko.eu/tag/android.html) / firefox-os (//k.japko.eu/tag/firefox-os.html) / boot.img (//k.japko.eu/tag/bootimg.html)

Introduction

Typical Android system implementing fastboot (this includes FirefoxOS devices) uses so called `boot.img` images to run the kernel. It consist of the kernel itself and a ramdisk that it used to populate root (`/`) filesystem. Because it's often useful to edit this filesystem (especially to modify `/default.prop` file), let's look how we can manipulate this image. I already briefly described how this can be done as part of my older post (//k.japko.eu/alcatel-otf-cwm.html) but now I'm going to describe more advanced and standalone (you don't need Android sources for it) tool.

I assume that you have an image dump and it's filename is `/tmp/boot.img` . You can dump it using commands like:

```
$ adb shell cat /dev/mtd/mtd0 >/mnt/sdcard/boot.img
$ adb pull /mnt/sdcard/boot.img /tmp/boot.img
```

You can read my older post (//k.japko.eu/alcatel-otf-cwm.html) to see how this can be done on Alcatel One Touch Fire.

Build abootimg tool

To do just about anything useful, you need proper tools. In this case, the tool is called `abootimg` . It's quite easy to install:

```
git clone https://git.gitorious.org/ac100/abootimg.git
cd abootimg
make
```

This will create `abootimg` executable which can be used right away (you don't have to copy it anywhere).

Extract boot.img content

In order to extract `boot.img` , use `-x` option followed with a path to the file. This will create few files in the current directory:

```
mkdir boot
cd boot
../abootimg -x /tmp/boot.img
```

Edit ramdisk content

`initrd.img` is a ramdisk file. It should be gzipped cpio file. To uncompress it, use following command:

```
mkdir initrd
cd initrd
cat ../initrd.img | gunzip | cpio -vid
```

Then edit the files you want, most probably `default.prop`. To recreate ramdisk image, use:

```
cd initrd
find . | cpio --create --format='newc' | gzip > ../myinitd.img
```

Repack boot.img

You may create new boot image using `--create` option. You have to specify config file (`-f` option), kernel image (`-k` option), and ramdisk image (`-r` option):

```
../abootimg --create myboot.img -f bootimg.cfg -k zImage -r myinitrd.img
```

If you extracted existing `boot.img` file, the config file was created for you. I explain this file later.

You may also change `boot.img` in place, using `-u` option combined with `-k` to change kernel image, `-r` to change ramdisk image, `-f` to change config file or `-c` to change specific config option only. For example to only change ramdisk and set `name` config option, you could use:

```
../abootimg -u /tmp/boot.img -c "name=rooted" -r myinitrd.img
```

Now you can boot this image with `fastboot boot myboot.img` command or flash it using `fastboot flash myboot.img`.

Configuration file

You can get some basic information from the image without extracting it with `-i` option:

```
$ ../abootimg -i ~/BUILD/firefoxos/B2G-hamachi/hamachi-backup/boot.img

Android Boot Image Info:

* file name = /home/k/BUILD/firefoxos/B2G-hamachi/hamachi-backup/boot.img

* image size = 4616192 bytes (4.40 MB)
  page size  = 2048 bytes

* Boot Name = ""

* kernel size       = 4274296 bytes (4.08 MB)
  ramdisk size      = 336854 bytes (0.32 MB)

* load addresses:
  kernel:       0x00c5c004
  ramdisk:      0x01f5c004
  tags:         0x00c54104

* cmdline = androidboot.hardware=qcom loglevel=1

* id = 0x19964b91 0xd35aa078 0x953b011d 0x2804aab4 0xf7a3e4b3 0x00000000 0x00000000 0x000
```

Most of this information is stored in a `bootimg.cfg` file and it’s used when recreating the image file. In some cases, you may want to change them. Here’s its example content:

```
$ cat bootimg.cfg
bootsize = 0xa00000
pagesize = 0x800
kerneladdr = 0xc5c004
ramdiskaddr = 0x1c54004
secondaddr = 0x1b54004
tagsaddr = 0xc54104
name =
cmdline = androidboot.hardware=qcom loglevel=1
```

- `bootsize` - size of the boot image; should be multiple of `pagesize`. It can never be smaller than the actual produced image file otherwise system won’t boot. You can remove this line and `abootimage` will calculate it for you. Of course, this can’t be bigger than your boot partition on the device.
- `pagesize` - it’s the size of a NAND page. You probably don’t want to change that.
- `kerneladdr`, `ramdiskaddr`, `secondaddr`, `tagsaddr` - specifies where in memory should each image be put by `fastboot` when flashing the image. You may want to change this if you want to change kernel in your image.

Note that those addresses are only used when you flash the image (`fastboot flash` command), if you boot it (`fastboot boot`) they will be calculated like this (typical `base` address is 0x10000000, it can be changed by `-b fastboot` parameter):

```
hdr->kernel_addr =  base + 0x00008000;
hdr->ramdisk_addr = base + 0x01000000;
hdr->second_addr =  base + 0x00F00000;
hdr->tags_addr =    base + 0x00000100;
```

I believe that the `base` address should be the same as `CONFIG_PHYS_OFFSET` in your kernel config.

- `name` is not used by bootloader but it can be displayed with `-i` option so you may want to use it to specify the purpose of each image, for example.
- `cmdline` contains command line passed to the kernel.

Social

My Github (http://github.com/kadamski/)

My Stackexchage (http://stackexchange.com/users/1733942/krzysztof-adamski)

Back to top