Forensic study of YAFFS2 filesystem

(Platform Usage)

redactor : buhtig@hashment.com

Table of Content

Chapter 1 : Launching QEMU virtual image

1.1 Foreword

But before booting the virtual image to work on it, let's check the init file so that it sets up the environment properly :

```
~ # more init
#!/bin/sh
exec > /dev/console 2>&1
echo "====[ Init start ]===="
mount -t proc none /proc
mount -t sysfs none /sys
mount -t tmpfs tmpfs /dev
echo "====[ Mdev starting ]===="
mdev -s
chmod 666 /dev/mtd*
echo "====[ Mounting ]===="

MY_DISK="/mnt/disk"
MY_YAFFS="/mnt/yaffs"
TEST_FILE="/mnt/disk/files_in_fs/"
RESULT="/mnt/disk/result"

mkdir -p $MY_DISK $MY_YAFFS
mount -t ext2 /dev/vda $MY_DISK
echo "====[ Modifying PATH ]===="
export PATH=/opt/bin:$PATH

echo "====[ Dropping to shell ]===="
exec /bin/sh
```

1.2 Booting the Image

Once booted, the virtual image appears as follows:

```
--[snip]--
[    1.778702] console [netcon0] enabled
[    1.778807] netconsole: network logging started
[    1.785635] Freeing unused kernel memory: 680k freed
[    1.821008] Write protecting the kernel read-only data: 12288k
[    1.826952] Freeing unused kernel memory: 692k freed
[    1.839732] Freeing unused kernel memory: 1608k freed
====[ Init start ]====
====[ Mdev starting ]====
[    1.918425] Refined TSC clocksource calibration: 3792.890 MHz.
[    1.918647] Switching to clocksource tsc
====[ Mounting ]====
[    1.946622] EXT2-fs (vda): warning: mounting unchecked fs, running e2fsck is recommended
[    1.948218] input: ImExPS/2 Generic Explorer Mouse as
/devices/platform/i8042/serio1/input/input2
====[ Modifying PATH ]====
====[ Dropping to shell ]====
/bin/sh: can't access tty; job control turned off
~ #
```

Concerning filesystem :

```
~ # mount
rootfs on / type rootfs (rw)
none on /proc type proc (rw,relatime)
none on /sys type sysfs (rw,relatime)
tmpfs on /dev type tmpfs (rw,relatime)
/dev/vda on /mnt/disk type ext2 (rw,relatime)
```

Concerning process :

```
~ # ps axf
PID   USER     TIME  COMMAND
    1 0         0:01 /bin/sh
    2 0         0:00 [kthreadd]
    3 0         0:00 [ksoftirqd/0]
    4 0         0:00 [kworker/0:0]
    5 0         0:00 [kworker/u:0]
    6 0         0:00 [migration/0]
    7 0         0:00 [cpuset]
    8 0         0:00 [khelper]
    9 0         0:00 [netns]
   10 0         0:00 [kworker/u:1]
  344 0         0:00 [sync_supers]
  346 0         0:00 [bdi-default]
  348 0         0:00 [kblockd]
  427 0         0:00 [ata_sff]
  438 0         0:00 [khubd]
  445 0         0:00 [md]
  454 0         0:00 [cfg80211]
  556 0         0:00 [rpciod]
  581 0         0:00 [kswapd0]
  649 0         0:00 [fsnotify_mark]
  663 0         0:00 [nfsiod]
  668 0         0:00 [crypto]
  785 0         0:00 [scsi_eh_0]
  788 0         0:00 [scsi_eh_1]
  800 0         0:00 [smflush]
  815 0         0:00 [mtdblock0]
  824 0         0:00 [mtdblock1]
  835 0         0:00 [mtdblock2]
  846 0         0:00 [kworker/0:2]
  877 0         0:00 [kpsmoused]
  949 0         0:00 ps axf
```

→ very minimal system

Concerning memory :

```
~ # free -m
             total         used         free       shared  buff/cache   available
Mem:           492           27          437            0           27           0
-/+ buffers/cache:           27          465
Swap:            0            0            0
```

Concerning disks :

```
~ # df
Filesystem           1K-blocks      Used Available Use% Mounted on
tmpfs                   251816         0    251816   0% /dev
/dev/vda                 20461       996     18441   5% /mnt/disk
```

1.3 Flashing the Simulated NAND

Reminder: the kernel was compiled with the MTD NANDSIM driver, which allows to emulate a NAND device.

To find where this driver is located, let's run the following command:

```
~ # cat /proc/mtd
dev:    size   erasesize  name
mtd0: 00400000 00020000 "mtdram test device"
mtd1: 04000000 00020000 "NAND simulator partition 0"       ← Here !!! in /dev/mtd1
mtd2: 01000000 00010000 "OneNAND simulator partition"
```

We can already mount the YAFFS2 file system on /dev/mtdblock1 (which is the block device associated with

/dev/mtd1).

However, by doing so:

```
~ # hexdump -C /dev/mtd1
00000000  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
04000000
```

...and comparing it with my base image:

```
~ # gzip -dc /mnt/disk/snapshot_00_empty.bin.gz | hexdump -C
00000000  ff ff ff ff ff ff ff ff  ff ff ff ff ff ff ff ff  |................|
*
04200000
```

It seems there is a difference. This is due to the fact that my image uses the following CHUNK parameters:

- DATA section size: 1024 bytes
- OOB (Out Of Band) section size: 64 bytes

Therefore, we need to re-flash the image before each test to ensure we always start from a known and consistent environment.

To do so, let's run the following commands:

```
        -- Erasing NAND --
#> flash_erase /dev/mtd1 0 0
Erasing 65536 Kibyte @ 0 -- 100 % complete


        -- Flashing NAND --
#> gzip -dc /mnt/disk/snapshot_00_empty.bin.gz | nandwrite -o /dev/mtd1 -
--[snip]--
Writing data to block 510 at offset 0x3fc0000
Writing data to block 511 at offset 0x3fe0000
Written 32768 blocks containing only 0xff bytes
Those block may be incorrectly treated as empty!
```

(→ ignore any errors and warnings.)

Then we mount the YAFFS2 partition on /mnt/yaffs:

```
~ # mount -t yaffs2 /dev/mtdblock1 /mnt/yaffs
[ 2003.124586] yaffs: dev is 32505857 name is "mtdblock1" rw
[ 2003.125104] yaffs: passed flags ""
```

The YAFFS partition is now mounted — we can verify this by:

```
~ # mount
rootfs on / type rootfs (rw)
none on /proc type proc (rw,relatime)
none on /sys type sysfs (rw,relatime)
tmpfs on /dev type tmpfs (rw,relatime)
/dev/vda on /mnt/disk type ext2 (rw,relatime)
/dev/mtdblock1 on /mnt/yaffs type yaffs2 (rw,relatime)
```

Let's play with it .....