ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 4, pp: (225-233), Month: October - December 2015, Available at: www.researchpublish.com

Data Acquisition Methodology for Forensic Investigation of Firefox OS

Mandar Jadhav[1], Prof. K.K Joshi[2]

[1]MTech Student, [2]Professor

[1,2]Department of Computer Engineering and IT, Veermata Jijabai Technological Institute, Mumbai, India

Abstract: **In 2012 Mozilla Corporation demonstrated an open source operating system named Firefox OS through its Boot2Gecko project. A year later, in 2013 the Firefox OS based phones were introduced in market. The widely accepted Android, iOS phones are there since many years, so research has been done on their digital investigation techniques and thus many forensic tools are available for such platforms. But, as the Firefox OS is still in its evolving phase not much research has been done on its forensic procedures, leading to lot of new opportunities and challenges for forensic investigators. Even the existing forensic tools does not fully support the Firefox OS based devices. Many tools detect the device as Android, the reason being the underlying Linux kernel in Firefox OS just like Android. Due to the increasing rate of mobile crimes the forensic analysis of this new, unique OS is urgently needed. The purpose of this paper is to present a method for acquiring forensic data from a mobile device running Firefox operating system. The paper provides a way to start acquisition process by using Android Debug Bridge before proceeding to identify the important files that should be used for analysis. Finally it shows how tools like qtADB, HexEditor and MOBILedit can be used in the acquisition phase of mobile forensics.**

Keywords: **Firefox OS; B2G; Mobile Forensics; Data Acquisition; Forensic investigation; Gecko.**

1. INTRODUCTION

In the era of hand-held devices many companies have started building their own mobile OS. Inspired from the gecko technology Mozilla introduced its Linux-based HTML5 powered Firefox OS [1]. The Firefox OS also named as B2G can be installed on few Android-compatible phones. ZTE released the first Firefox OS smartphone in 2013. Apart from ZTE other vendors of Firefox OS based devices are Alcatel, Symphony, Zen, Intex, Spice, Geeksphone and Cherry Mobile. Firefox OS implements a four layered architecture comprising of Gaia, Gecko and Gonk layers sitting on top of Mobile device. Gecko, an implementation of technologies like HTML5, CSS, and JavaScript provides framework for execution. The applications are connected to hardware for accessing features of the device through Web APIs. Gonk is an abstraction layer based on Android Open Source Project (AOSP) which comprises of Linux kernel, firmware, libraries and drivers. Gaia is the user interface implemented using HTML, CSS and javascript, while mobile device is the phone that runs Firefox OS. Most of the data needed for forensic analysis is situated at the Gecko layer. The challenge is to acquire this data. Mobile forensic acquisition process can be classified into three types; logical, physical and manual [3]. In physical acquisition information can be extracted from flash memory and bit-by-bit copy of complete file system is made for analysis. Logical acquisition is carried out with help of manufacturers API and other tools for recovery of files on mobile device. Manual acquisition involves using the user interface of mobile device to examine the contents. All the three categories have their own benefits and limitations. According to Grispos , physical acquisition can retrieve data from unallocated space, but logical acquisition can only recover user data, whereas manual acquisition can be used as means to verify both logical and physical acquisition [6]; In this paper an approach for logical and physical acquisition is suggested. We will be identifying all partitions and the file system structure. Android Debug Bridge (ADB), Google USB driver will be used to connect the phone to host system. As per our knowledge the existing forensic tools are not working perfectly with Firefox OS. The device is detected as an Android phone due to similarity of kernel. We used a graphical version of ADB called qt ADB to identify and acquire important files and analyzed them in hexEditor and MOBILedit! to deduce important results. Finally we recommend the use of dd command or netcat to create image of the forensic data that can be used for analyzing forensic artifacts.

The aim of this paper is to present a method for acquiring forensic data from Firefox OS using ADB without making any changes to the device. The paper is organized as follows; Section (II) will give the overview of existing approaches. Section (III) will put light on the boot process. Section (IV) introduces our suggested acquisition methodology, partition structure and use of ADB commands. Section (V) will show the results of experiments carried out on Alcatel One Touch Fire C device using qtADB, HexEditor and MOBILedit! tool. Section (VI) will give conclusion and the intended future work. All the References are mentioned at the end of this paper.

2. OVERVIEW OF EXISTING MOBILE FORENSICS APPROACHES

Among the many available smart mobile devices, the market is mainly dominated by Android, iOS, Blackberry and Windows based phones. The distinct features of these operating systems gave rise to many techniques for acquisition and examination of data. Every investigation is specific to a device and it is influenced by the operating system, hardware, file system and partitions.

Operating system is the major factor in acquisition process as it guides the mode of access. Android being an open source OS gives terminal-level access but such access is restricted in iOS. Digital forensic acquisition needs manual effort along with the forensic tools. Many forensic tools and platforms like MOBILedit!, FTK, Kali Linux, etc. are available. All tools have their own limitations and benefits. So it is imperative for investigators to have good understanding of forensic tools.

Classifying Forensic Tools:

The mobile forensic tools are classified on basis of the following five levels as follows:

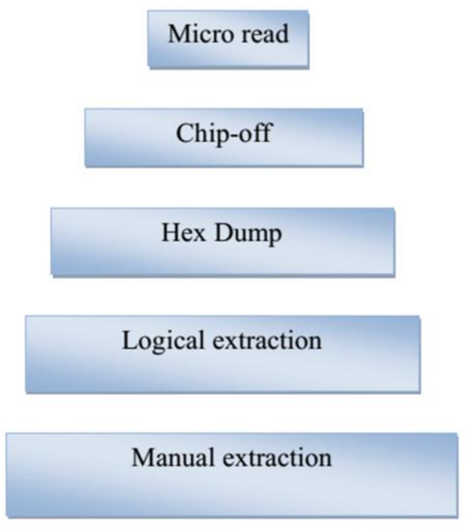


Fig. 1: Levels of acquisition

In manual extraction data is viewed by using the touchscreen interface or keypad. Photos are clicked and documented. However, only limited data can be acquired. In Logical acquisition the phone is connected using USB, Wifi or Bluetooth; to the host system where forensic tools are installed. All the partitions are analyzed to create an image of evidence. Deleted information cannot be acquired using this approach. In hex dump method forensic tools generate a raw image of memory dump. The binary image is then analyzed. Maximum evidence can be acquired from this method. The expensive and risky Chip-off method acquires data by removing the memory chip and placing it inside a chip reader. Machine level knowledge is required for the heating and de-soldering of the memory chip. This method can damage the chip. The Joint Test Action Group (JTAG) process which establishes connectivity using Test Access Ports (TAPs) is usually followed. In Micro Read a microscope is used to determine the gates and its status. It is a time consuming process.

Rules of Evidence:

- Authenticity: The analysis should be carried in relevant way and origin should be well explained.
- Completeness: The evidence should be clear and complete to depict the entire story.
- Reliability: The evidence should be collected using reliable tools.
- Admissible : The gathered evidence should be admissible, so it should not be acquired using illegal methods
- Believable: The evidence should be clearly explained and its integrity must be preserved so that it can be believed.

Limiting Factors:

The following are the critical factors that can be roadblocks in investigation process :

- Lack of resources: Non-availability of hardware resources like USB cables, chargers, batteries and software resources like device compatible USB drivers makes it difficult to proceed.
- Device state: The background processes may cause modifications of data.
- Dynamic data: The evidence highly dynamic in nature can be altered easily if not handled carefully.
- Device reset: Device can be reset accidently, thus wiping out important data.
- Passcode : It is possible that the device uses passcode for authentication, data can be damaged while bypassing such barrier.
- Malicious programs: Malwares like viruses and Trojans may reside on the device leading to faulty analysis.

3. FIREFOX OS BOOTUP PROCESS

Boot procedure:

As shown in the figure below, the booting starts from bootloaders in the Kernel and then it moves to native code space to init process. After this it enters the user space to execute B2G and Gecko before moving to browse layer to run the system app followed by window manager. Finally the homescreen application executes inside the Gecko runtime. Most devices use the fastboot protocal.

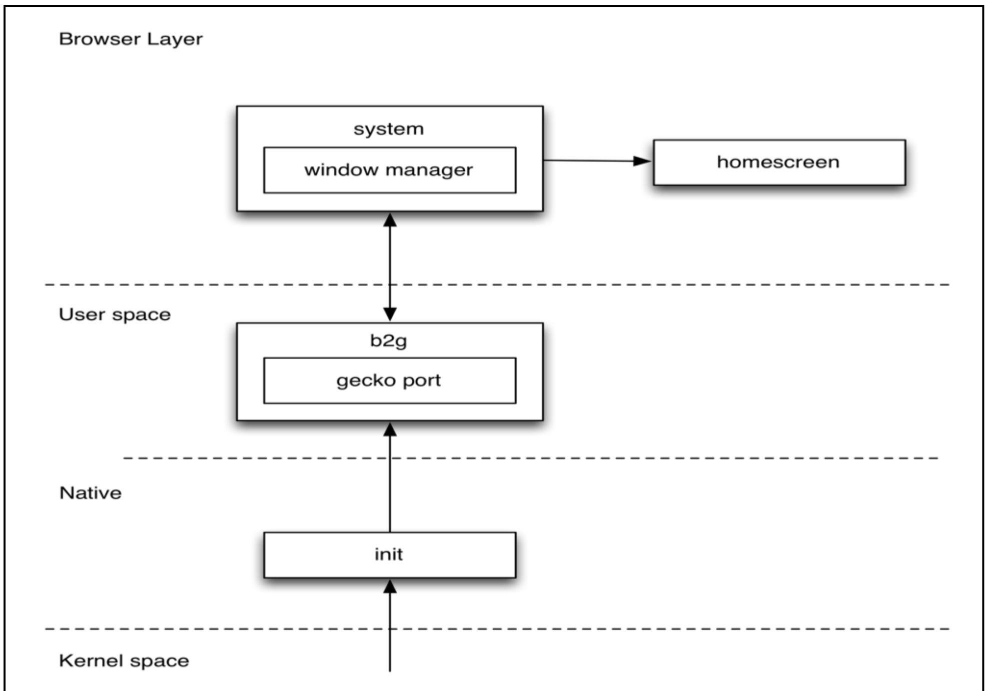


Fig. 2: Boot process

Userspace processes:

- init: This process mounts the file systems and discovers services. It starts the b2g process and manages other processes.
- b2g: This is the high-privileged primary process that establishes communication with cameras, GPS, modem, other sensors.
- netd: Network interfaces are configured using netd process.
- rild: This daemon is the hardware level implementation of Radio Interface Layer(RIL).
- rilproxy: It is the rild client acting as a passive proxy between b2g and rild process.
- Mediaserver: This process controls the playback of audio and video using remote procedure calls.
- dbus-daemon : Bluetooth needs a message bus called D-bus for communicating with other devices, dbus-daemon implements this bus.

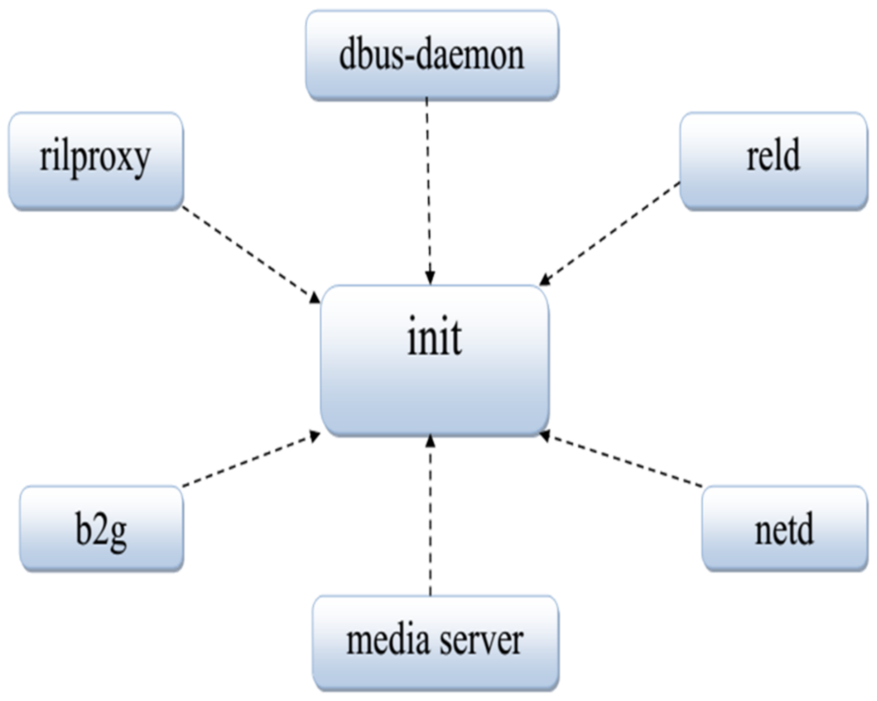


Fig. 3 : processes spawned by init

4. ACQUISITION METHODOLOGY

Our proposed methodology is carried out in two phases. First we need to establish the connection between host and the device before studying the partitions, followed by identifying the important sources of information. Finally the relevant data is acquired. Our approach starts with knowledge preparation. Before starting any forensic investigation process the investigator should have sound knowledge about the forensic tools, operating system architecture, storage partitions. So information about Firefox OS is gathered first. The next step is to define and setup the forensic environment. Here we choose the host device with suitable configuration, install the selected forensic software and the USB drivers for mounting the device.

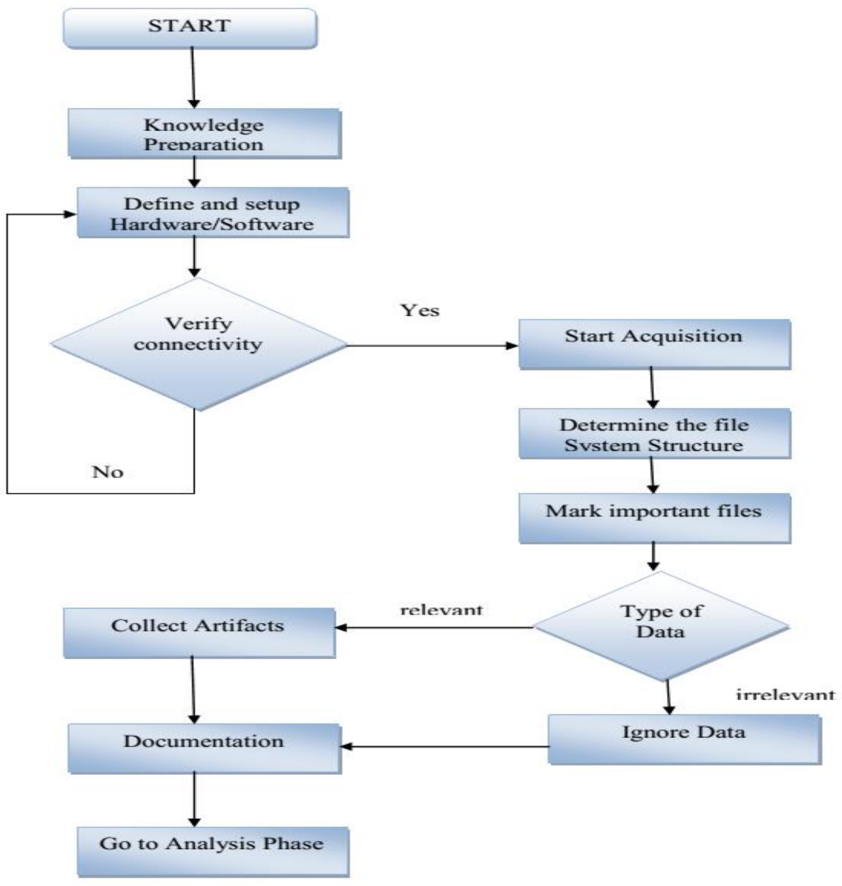


Fig. 4 : Proposed Acquisition Methodology

If everything goes well and the device is detected we can move to next step. Once the connectivity is verified we can start the acquisition process. We use the Android Debug Bridge from Android SDK to check the available partitions. The next step is to mark the important user and system files that may contain meaningful evidence. The entire data does not have much forensic value, it only increases the time of acquisition, and therefore we need to focus only on the relevant data. We create a copy of the important files on host device. Next, all the steps should be documented properly for acceptability. The acquired data can then be fed to Analysis phase.

5. EXPERIMENTAL RESULTS FROM ACQUISITION PHASE

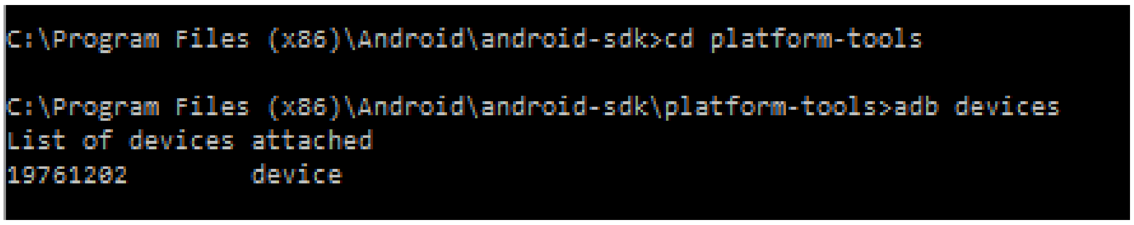
In our acquisition process we used Firefox OS device manufactured by Alcatel, model name One Touch Fire C. The TABLE I gives the full phone specification.

TABLE I

HARDWARE	DETAIL
PLATFORM	OS – Firefox OS 1.3 Chipset – Spreadtrum 6821 CPU – 1 GHZ
MEMORY	Internal – 256mb ROM, 128mb RAM External – microSD upto 32 GB
BATTERY	Li-Ion 1000mAh
CONNECTIVITY	AWLAN – Wi-fi 802.11 b/g/n Bluetooth – v2.1 Radio – FM radio USB – microUSB 2.0
DIMENSIONS	Width – 112.2mm Height – 62mm Thickness – 12mm

The forensic environment is setup on a Dell Inspiron 15R series Windows 10 booted laptop. Android SDK , qtADB, HexEditor and MOBILedit! was installed on the host. As the device specific USB drivers were not available we used Google USB driver from Android SDK and the Universal USB Driver to connect the device to host. Disable the USB storage on the phone and connect it to the host. To check whether the device is connected, open the command prompt, navigate to platform-tools folder located inside the Android SDK and run the following command:

adb devices



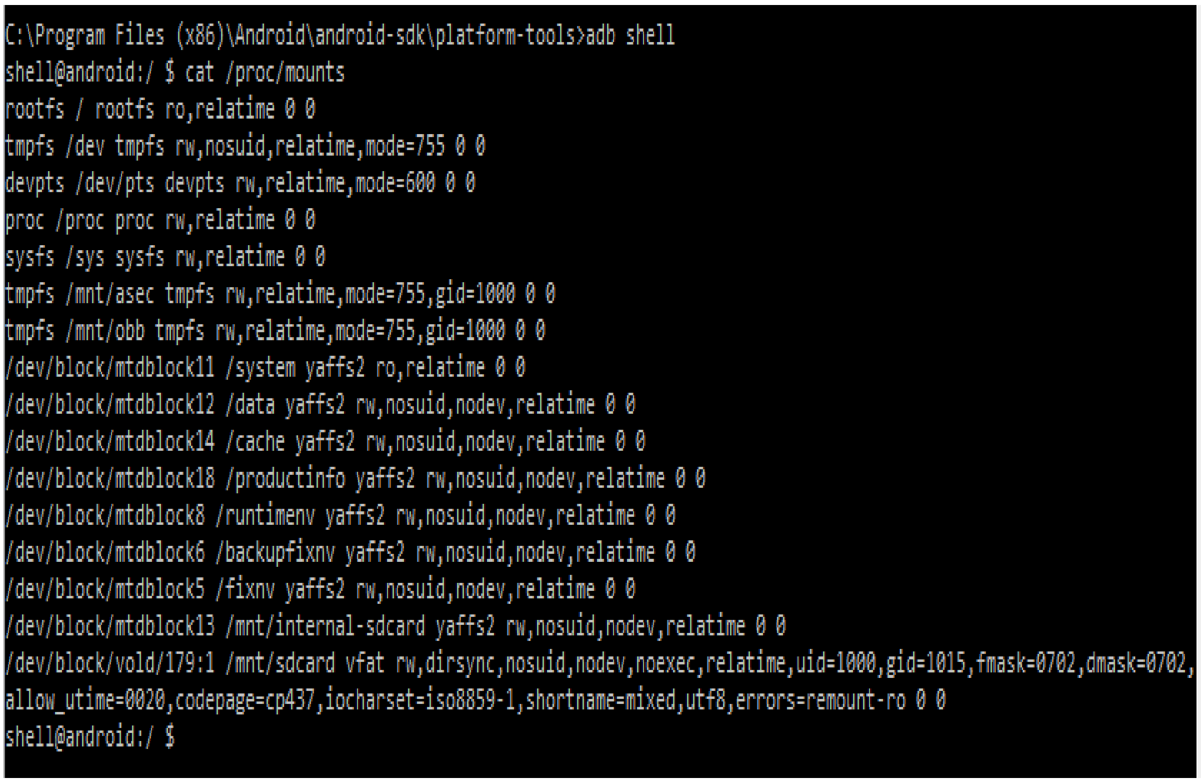
```
C:\Program Files (x86)\Android\android-sdk>cd platform-tools

C:\Program Files (x86)\Android\android-sdk\platform-tools>adb devices
List of devices attached
19761202        device
```

Fig 5: Firefox OS device connected successfully.

Then we found the file system structure and the various file formats supported by openining an ADB shell and using the following command :

cat /proc/mounts



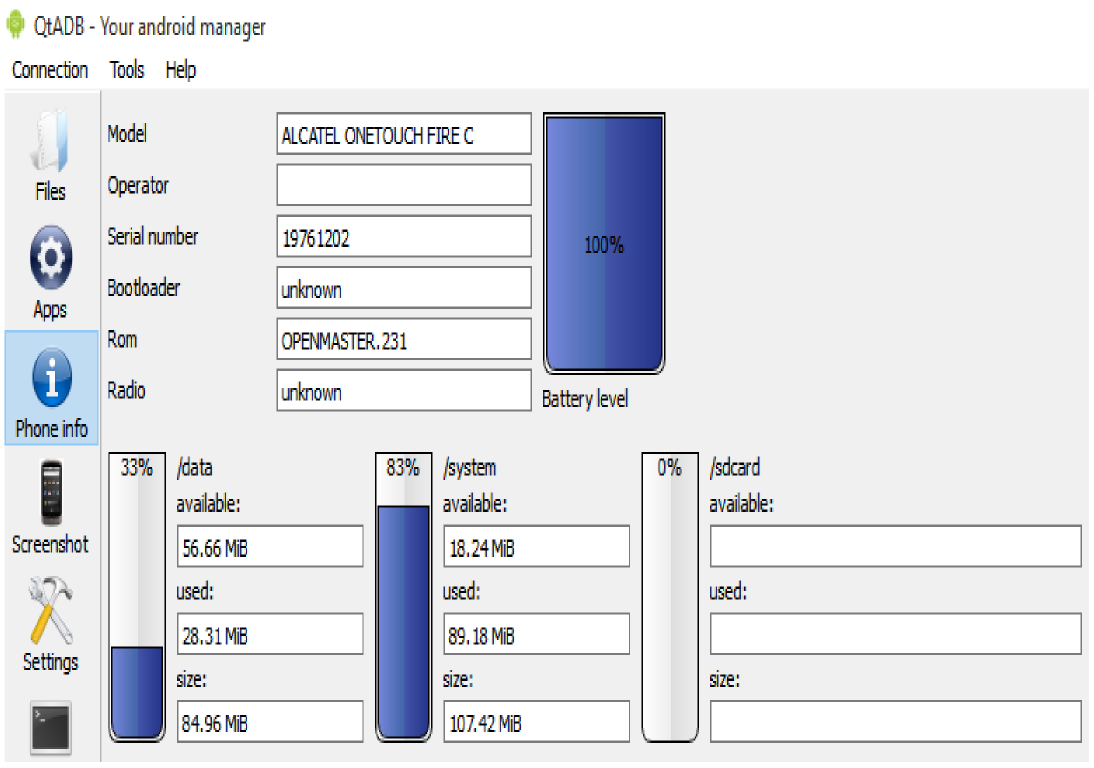
```
C:\Program Files (x86)\Android\android-sdk\platform-tools>adb shell
shell@android:/ $ cat /proc/mounts
rootfs / rootfs ro,relatime 0 0
tmpfs /dev tmpfs rw,nosuid,relatime,mode=755 0 0
devpts /dev/pts devpts rw,relatime,mode=600 0 0
proc /proc proc rw,relatime 0 0
sysfs /sys sysfs rw,relatime 0 0
tmpfs /mnt/asec tmpfs rw,relatime,mode=755,gid=1000 0 0
tmpfs /mnt/obb tmpfs rw,relatime,mode=755,gid=1000 0 0
/dev/block/mtdblock11 /system yaffs2 ro,relatime 0 0
/dev/block/mtdblock12 /data yaffs2 rw,nosuid,nodev,relatime 0 0
/dev/block/mtdblock14 /cache yaffs2 rw,nosuid,nodev,relatime 0 0
/dev/block/mtdblock18 /productinfo yaffs2 rw,nosuid,nodev,relatime 0 0
/dev/block/mtdblock8 /runtimenv yaffs2 rw,nosuid,nodev,relatime 0 0
/dev/block/mtdblock6 /backupfixnv yaffs2 rw,nosuid,nodev,relatime 0 0
/dev/block/mtdblock5 /fixnv yaffs2 rw,nosuid,nodev,relatime 0 0
/dev/block/mtdblock13 /mnt/internal-sdcard yaffs2 rw,nosuid,nodev,relatime 0 0
/dev/block/vold/179:1 /mnt/sdcard vfat rw,dirsync,nosuid,nodev,noexec,relatime,uid=1000,gid=1015,fmask=0702,dmask=0702,
allow_utime=0020,codepage=cp437,iocharset=iso8859-1,shortname=mixed,utf8,errors=remount-ro 0 0
shell@android:/ $
```

Fig 6 : All Partitions

We can easily identify that internal storage files are present on *mtdblock13*. So we mark this file as relevant data. Similarly other important areas of interest like user data on *mtdblock12* can be identified. The system partitions in Firefox OS are quite unique as compared to other mobile operating systems. A part of code snippet of internal partition is given below:

```
        }, static struct mtd_partition partition_info[]=
{
        {
                .name =         "ADNP boot",
                .offset = 0,
                .size =         0xf0000,
        },
        {
                .name =         "ADNP system BIOS",
                .offset = MTDPART_OFS_NXTBLK,
                .size =         0x10000,
#ifdef DNPC_BIOS_BLOCKS_WRITEPROTECTED
                .mask_flags =   MTD_WRITEABLE,
#endif
        },
        {
                .name =         "ADNP file system",
                .offset = MTDPART_OFS_NXTBLK,
                .size =         0x2f0000,
        },
        {
                .name =         "ADNP system BIOS entry",
                .offset = MTDPART_OFS_NXTBLK,
                .size =         MTDPART_SIZ_FULL,
#ifdef DNPC_BIOS_BLOCKS_WRITEPROTECTED
                .mask_flags =   MTD_WRITEABLE,
#endif
};
#define NUM_PARTITIONS ARRAY_SIZE(partition_info)
static struct mtd_info *mymtd;
static struct mtd_info *lowlvl_parts[NUM_PARTITIONS]; static struct mtd_info *merged_mtd;
```

The copy of this relevant data can be made using DD or NETCAT utilities. After identifying important blocks we moved to graphical version of ADB called qtADB. Enable the USB Storage before connecting to this software.


QtADB - Your android manager
Connection Tools Help
Files
Apps
Phone info
Screenshot
Settings

Model: ALCATEL ONETOUCH FIRE C
Operator:
Serial number: 19761202
Bootloader: unknown
Rom: OPENMASTER.231
Radio: unknown
Battery level: 100%

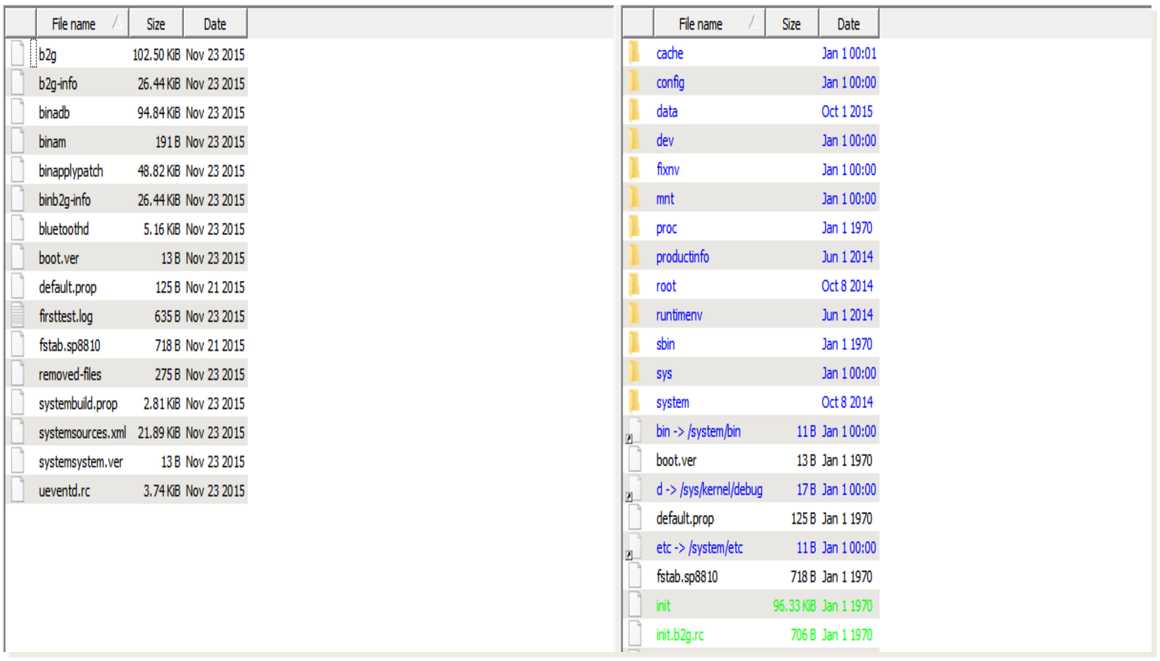
33% /data
available: 56.66 MiB
used: 28.31 MiB
size: 84.96 MiB

83% /system
available: 18.24 MiB
used: 89.18 MiB
size: 107.42 MiB

0% /sdcard
available:
used:
size:

Fig. 7 Phone information

In qtADB all the files and folders of mobile device are shown on right side and host machine is shown on left side. All the important files were copied to host system and we analyzed the *default.prop* file using hexEditor.

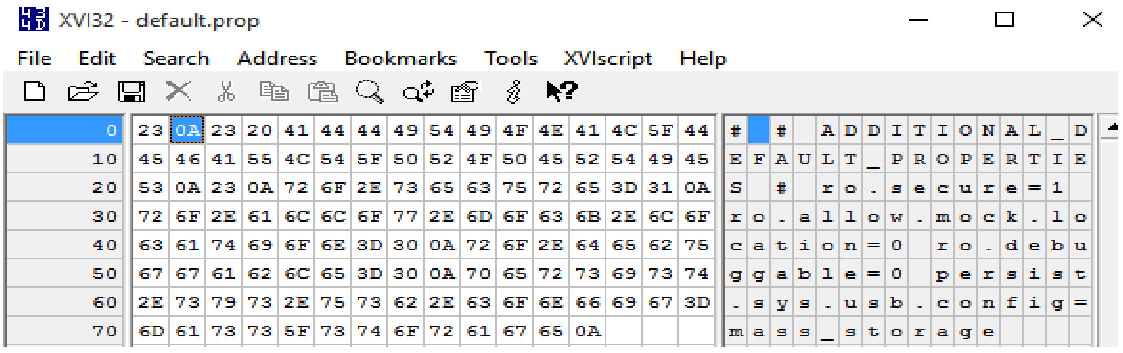


File name	Size	Date
b2g	102.50 KiB	Nov 23 2015
b2g-info	26.44 KiB	Nov 23 2015
binadb	94.84 KiB	Nov 23 2015
binam	191 B	Nov 23 2015
binapplypatch	48.82 KiB	Nov 23 2015
binb2g-info	26.44 KiB	Nov 23 2015
bluetoothd	5.16 KiB	Nov 23 2015
boot.ver	13 B	Nov 23 2015
default.prop	125 B	Nov 21 2015
firsttest.log	635 B	Nov 23 2015
fstab.sp8810	718 B	Nov 21 2015
removed-files	275 B	Nov 23 2015
systembuild.prop	2.81 KiB	Nov 23 2015
systemsources.xml	21.89 KiB	Nov 23 2015
systemsystem.ver	13 B	Nov 23 2015
ueventd.rc	3.74 KiB	Nov 23 2015

File name	Size	Date
cache		Jan 1 00:01
config		Jan 1 00:00
data		Oct 1 2015
dev		Jan 1 00:00
fixnv		Jan 1 00:00
mnt		Jan 1 00:00
proc		Jan 1 1970
productinfo		Jun 1 2014
root		Oct 8 2014
runtimenv		Jun 1 2014
sbin		Jun 1 1970
sys		Jan 1 00:00
system		Oct 8 2014
bin -> /system/bin	11 B	Jan 1 00:00
boot.ver	13 B	Jan 1 1970
d -> /sys/kernel/debug	17 B	Jan 1 00:00
default.prop	125 B	Jan 1 1970
etc -> /system/etc	11 B	Jan 1 00:00
fstab.sp8810	718 B	Jan 1 1970
init	96.33 KiB	Jan 1 1970
init.b2g.rc	706 B	Jan 1 1970

Fig. 8 : Acquired files

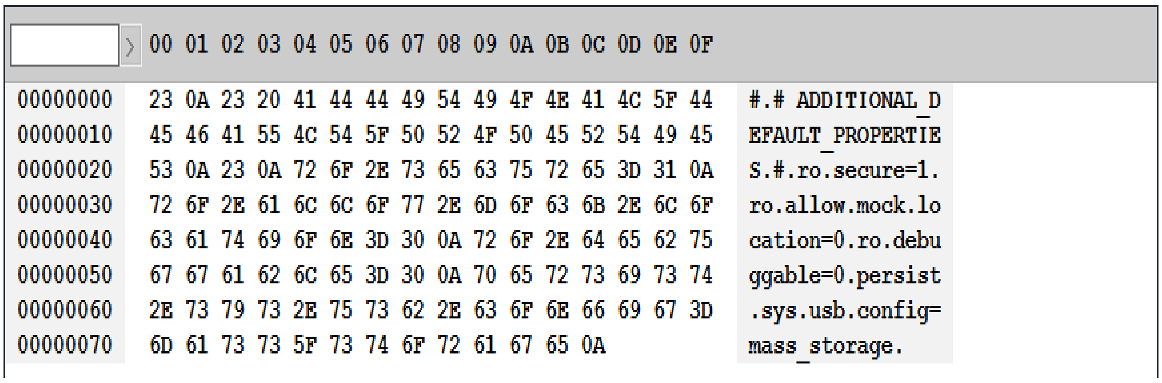
By analyzing the *default.prop* we got information about the two variables *ro.secure* and *ro.debuggable* set to 1 and 0 respectively. This tells us that the phone is not rooted.



```
 0  23 0A 23 20 41 44 44 49 54 49 4F 4E 41 4C 5F 44   #   #   A D D I T I O N A L _ D
10  45 46 41 55 4C 54 5F 50 52 4F 50 45 52 54 49 45   E F A U L T _ P R O P E R T I E
20  53 0A 23 0A 72 6F 2E 73 65 63 75 72 65 3D 31 0A   S   #   r o . s e c u r e = 1
30  72 6F 2E 61 6C 6C 6F 77 2E 6D 6F 63 6B 2E 6C 6F   r o . a l l o w . m o c k . l o
40  63 61 74 69 6F 6E 3D 30 0A 72 6F 2E 64 65 62 75   c a t i o n = 0   r o . d e b u
50  67 67 61 62 6C 65 3D 30 0A 70 65 72 73 69 73 74   g g a b l e = 0   p e r s i s t
60  2E 73 79 73 2E 75 73 62 2E 63 6F 6E 66 69 67 3D   . s y s . u s b . c o n f i g =
70  6D 61 73 73 5F 73 74 6F 72 61 67 65 0A            m a s s _ s t o r a g e
```

Fig. 9 : HexEditor output

For verifying our results we used another forensic tool called MOBILedit!. The device was detected as Android. We were able to acquire the *default.prop* and analyze its hex dump.



```
          00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000  23 0A 23 20 41 44 44 49 54 49 4F 4E 41 4C 5F 44   #.# ADDITIONAL_D
00000010  45 46 41 55 4C 54 5F 50 52 4F 50 45 52 54 49 45   EFAULT_PROPERTIE
00000020  53 0A 23 0A 72 6F 2E 73 65 63 75 72 65 3D 31 0A   S.#.ro.secure=1.
00000030  72 6F 2E 61 6C 6C 6F 77 2E 6D 6F 63 6B 2E 6C 6F   ro.allow.mock.lo
00000040  63 61 74 69 6F 6E 3D 30 0A 72 6F 2E 64 65 62 75   cation=0.ro.debu
00000050  67 67 61 62 6C 65 3D 30 0A 70 65 72 73 69 73 74   ggable=0.persist
00000060  2E 73 79 73 2E 75 73 62 2E 63 6F 6E 66 69 67 3D   .sys.usb.config=
00000070  6D 61 73 73 5F 73 74 6F 72 61 67 65 0A            mass_storage.
```

Fig. 10: MOBILedit! hex dump of default.prop

From the output we deduced that the device was not rooted as the hex dump was similar to the one given by qtADB. Thus we successfully verified our results of analysis.

6. CONCLUSION

The Firefox OS is one of its kind, designed to harness the power of web. It has lot of potential to succeed in mobile market. As it has limited restrictions it will attract hackers all around the world. So research needs to be done for its sound forensic investigation. We proposed a methodology for carrying out the acquisition process starting with establishing the connectivity using ADB. We were successful in acquiring relevant user and system data using qtADB. As the device was not rooted we had some restrictions with regards to system files. Analysis of defaut.prop file gave us same results in both qtADB and MOBILedit!. Later on we will focus on creating bit-by-bit copy of important blocks using DD command. We will try to create the image directly on host system using NETCAT. In future, we will work on acquiring and analyzing data from dual booted Android-Firefox OS device. In this paper we focused mainly on the acquisition process and explained the possible ways of proceeding with the analysis.

REFERENCES

[1] Mohd Najwadi Yusoff, Ramlan Mahmod, Ali Dehghantanha, Mohd Taufik Abdullah,"An Approach for Forensic Investigation in Firefox OS", IEEE, CyberSec, May-2014.

[2] Mohd Najwadi Yusoff, Ramlan Mahmod, Ali Dehghantanha, Mohd Taufik Abdullah,Mobile, "Forensic Data Acquisition in Firefox OS", IEEE, CyberSec ,2014.

[3] K. Barmpatsalou, D. Damopoulos, and G. Kambourakis, "A critical review of 7 years of Mobile Device Forensics," Digit. Investig., vol. 10, no. 4, pp. 323–349, 2013.

[4] S.-W. Chen, C.-H. Yang, and C.-T. Liu, "Design and Implementation of Live SD Acquisition Tool in Android Smart Phone," Fifth International Conference on Genetic and Evolutionary Computing, 2011.

[5] Mozilla Developer Network, "Firefox OS architecture," https://developer.mozilla.org/enUS/docs/Mozilla/ Firefox_OS/Platform/Architecture. 07-May-2013.

[6] G. Grispos, T. Storer, and W. B. Glisson, "A comparison of forensic evidence recovery techniques for a windows mobile smart phone," Digital Investigation, vol. 8, no. 1, pp. 23–36, Jul. 2011.

[7] PACKT Publishing, "Introduction to Mobile Forensics", https://www.packtpub.com/books/content/introduction-mobile-forensics.

[8] GSM Arena, "Alcatel Fire C 2G", http://www.gsmarena.com/alcatel_fire_c_2g-6712.php.

[9] PACKT Publishing, "Extracting data Physically with DD", https://www.packtpub.com/books/content/extracting-data-physically-dd.

[10] MOBILedit Forensics, "Complete data extraction", http://www.mobiledit.com/forensics.

[11] Android Central, "Ten Basic Android terminal commands", http://www.androidcentral.com/android-201-10-basic-terminal-commands-you-should-know.