New Acquisition Method Based On Firmware Update Protocols For Android Smartphones

By

Seung Jei Yang, Jung Ho Choi, Ki Bom Kim and Tae Joo Chang

From the proceedings of

The Digital Forensic Research Conference

DFRWS 2015 USA

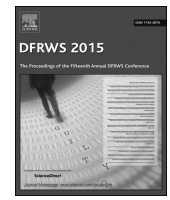
Philadelphia, PA (Aug 9th - 13th)

Digital Investigation 14 (2015) S68–S76

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin



DFRWS 2015 US

New acquisition method based on firmware update protocols for Android smartphones

Seung Jei Yang*, Jung Ho Choi, Ki Bom Kim, Taejoo Chang

The Affiliated Institute of ETRI, P.O. Box 1, Yuseong, Daejeon, 305-600, Republic of Korea

Keywords:
Android forensics
Android physical acquisition
Firmware update protocol
Flash memory read command
Bootloader

A B S T R A C T

Android remains the dominant OS in the smartphone market even though the iOS share of the market increased during the iPhone 6 release period. As various types of Android smartphones are being launched in the market, forensic studies are being conducted to test data acquisition and analysis. However, since the application of new Android security technologies, it has become more difficult to acquire data using existing forensic methods. In order to address this problem, we propose a new acquisition method based on analyzing the firmware update protocols of Android smartphones. A physical acquisition of Android smartphones can be achieved using the flash memory read command by reverse engineering the firmware update protocol in the bootloader. Our experimental results demonstrate that the proposed method is superior to existing forensic methods in terms of the integrity guarantee, acquisition speed, and physical dump with screen-locked smartphones (USB debugging disabled).

Introduction

The Android OS accounted for approximately 84% of the market share in the third quarter of 2014 (Smartphone OS Market Share Q3 2014). The size of the Android smartphone market now exceeds that of the PC market and various technologies are emerging continuously for personal and business use (Bring your own device, 2014). This trend is increasing the importance of Android forensics, where research into the physical acquisition of flash memory is particularly necessary to aid the recovery and analysis of deleted files.

The existing Android physical acquisition methods have the following problems.

First, most forensic tools acquire data from smartphones by exploiting Android kernel vulnerabilities or using custom images (Rooting (Android OS), 2014; Vidas et al., 2011). However, these vulnerabilities have been patched as the Android OS has been upgraded and physical memory dumping is not supported in the latest OS until a new vulnerability can be found. Furthermore, the recent applications of security technologies (Secure boot, 2014; Samsung KNOX, 2014) make it even harder to acquire data from smartphones.

Second, the dump method based on changing the custom recovery image (Son et al., 2013) is the only approach that takes into account the integrity of the user data. However, this method cannot guarantee the integrity of the entire flash memory dump because it also flashes the custom recovery image.

Third, existing forensic tools use the Android Debug Bridge (ADB) protocol to acquire data from smartphones. Thus, it is difficult to acquire data when smartphones are locked by a pattern or user password (USB debugging disabled).

* Corresponding author. Tel.: +82 42 870 2343, +82 10 2321 4588; fax: +82 42 870 2222.
E-mail addresses: sjyang@nsr.re.kr, sjyangub@dreamwiz.com (S.J. Yang).

http://dx.doi.org/10.1016/j.diin.2015.05.008

In order to address these problems, we propose a new physical acquisition method based on analyzing the firmware update protocols of Android smartphones.

Related work

Software (S/W)-based and hardware (H/W)-based acquisition methods are mainly used to acquire data from Android smartphones.

The S/W-based acquisition methods are divided into logical acquisition and physical acquisition.

Logical acquisition methods acquire user data stored on a smartphone via ADB Backup (Android Backup Extractor, 2014) or Content Provider (Hoog, 2011). However, this method only retrieves stored files such as the call history and pictures, and it cannot recover deleted files.

Physical acquisition methods extract the overall data directly from the smartphone's flash memory after connecting a USB cable. To perform a physical dump of the flash memory, the rooting process required to obtain an administrative privilege must be performed first. The rooting-based acquisition studies were introduced in the scholarly works (Hoog, 2009; Lessard and Kessler, 2010). These studies discussed Android forensics, including acquisition methods that root the HTC smartphones and use ADB shell. However, these methods can only be used to acquire data when the USB debugging mode is enabled. Commercial forensic tools (Oxygen Forensics, 2014; AccessData MPE+, 2014; MSAB XRY, 2015) also use this method. However, because the rooting exploitation process is executed after the smartphone is booted; the integrity is damaged whenever data is acquired. In addition, existing rooting vulnerabilities are patched whenever the Android OS is updated with a new version; hence, a new rooting technique must be found whenever the Android OS is updated.

Cellebrite UFED 4PC (2015) basically supports an ADB physical memory dump via rooting exploitation, while some Samsung models support physical memory dumping via a custom bootloader. However, this method has the problem that each model has to upload a different bootloader rather than uploading a common loader for physical memory dump. Because the physical dump is not supported in some models in the same Galaxy series, this method is not regarded as stable.

An acquisition method based on changing the custom recovery image has been studied in the scholarly works (Vidas et al., 2011; Son et al., 2013). Use of the custom recovery image guarantees the integrity of the user data. However, this method cannot guarantee the integrity of the entire flash memory dump because it also flashes the custom recovery image. The acquisition method must have USB debugging enabled because it uses the ADB shell protocol to acquire data from smartphones. However, because USB debugging is usually disabled, this method is not applicable if the pattern lock or user password is set. Moreover, the Secure Boot and Samsung KNOX technologies applied recently to Android place restrictions on the flashing of custom images, which will make it difficult when using this acquisition method in the future.

Flasher tools (RIFF box, 2014; ORT tool, 2014; Z3X box, 2014) are also used to extract data from mobile devices. However, the main function of these tools is to fix the bricked phones that have S/W damages. So these tools are not considered as general forensic tools.

Representative H/W-based acquisition methods include JTAG-based acquisition (Kim et al., 2008; Breeuwsma et al., 2007) and Chip-off-based acquisition (Jovanovic, 2012). The JTAG-based acquisition method extracts data from the flash memory using the JTAG debug interface on the smartphone's PCB board. The Chip-off-based method physically removes flash memory chips from the PCB board of smartphones and acquires the raw data of the flash memory. The JTAG-based acquisition method is problematic because not all smartphones support JTAG and it takes a long time to acquire data. The Chip-off-based acquisition method is utilized in limited situations because it separates the flash memory.

Also, there were various studies conducted on Android forensics. As the capacity of the main memory grows in Android smartphones, the forensic research for volatile memory was introduced (Sylve et al., 2012). The analysis of social networking applications (Mutawa et al., 2012), the prototype enterprise monitoring system for Android smartphones (Grover, 2013), and the Kindle forensics (Hannay, 2011; Iqbal et al., 2013) were also studied.

Background

Flash memory is used mainly to store data on smartphones. Because the flash memory is small and it can store a large amount of data, it is used widely in embedded devices such as smartphones and feature phones. Recently embedded Multi-Media Card (eMMC) that integrates NAND flash and a controller into a package has been used; this manages stored data by efficiently using the EXTended file system 4 (EXT4). Moreover, it mounts and operates partitions such as BOOT, RECOVERY, SYSTEM, and USERDATA.

An administrative privilege must be obtained before the physical dump of the entire flash memory. Thus, a custom image is overwritten in the BOOT partition in order to obtain the administrative privilege, and an app (SuperUser.apk) and a binary (/system/su) are installed in the SYSTEM partition. In addition, the administrative privilege is obtained via rooting exploitation in the recovery mode or by exploiting vulnerabilities in the Android OS.

In general, Google's FASTBOOT (Android software development-fastboot, 2014) is used for flashing a custom image. Each manufacturer provides their own firmware update programs (Samsung Kies, 2014; Samsung Odin, 2014; LG Software & tools Download, 2014; Pantech Self-Upgrade, 2014; HTC Sync Manager, 2014; Sony PC Companion, 2014; Xiaomi Download MiFlash for Xiaomi Smartphone, 2015) in order to prevent the simple flashing through FASTBOOT provided by Google, and they do not publish a protocol so only the original firmware can flash. The firmware update process runs only when smartphones enter a special mode called firmware update, or the download mode. Only the bootloader and USB function can operate in this mode and a new system firmware can be flashed.

Firmware update processes and commands can be analyzed by reverse engineering the bootloader and firmware update program using a tool such as IDA Pro (Hex-Rays, 2015).

Android physical acquisition based on firmware update protocols

From a forensic viewpoint, flash memory that contains user data is the main target during data acquisition. This process requires a physical acquisition method to acquire the entire flash memory, rather than a logical acquisition method.

During the firmware update process, new firmware is flashed in the flash memory in order to update the Android OS or patch S/W problems. A firmware update protocol is the only way to access the flash memory directly through S/W, and thus we can derive a new physical memory acquisition method by analyzing the commands used in the firmware update process. There was no research that analyzes firmware update protocols in order to solve the problems of existing forensic acquisition tools so far. In the present study, we analyzed the firmware update protocols used by LG, Pantech, and Samsung smartphones. In the LG and Pantech models, we found that not only commands for direct access to flash memory and the write commands for flashing but also the read commands for flash memory dumping were preserved intact. In Samsung models, we confirmed that there were the read commands for flash memory dumping, but the actual dump code had been removed. Based on these analytical results, we propose a new physical acquisition method for Android smartphones.

Firmware update protocol

If an Android smartphone is turned on or reset, the ROM is executed from address 0 and initialization processes are executed, including CPU configuration. Next, the bootloader is loaded onto the memory, and H/W is initialized and configured, such as NAND and USB. Bootloader execution passes through several stages, such as Initial BootLoader (IBL), Primary BootLoader (PBL), Secondary BootLoader (SBL), and a firmware update protocol is implemented in the SBL bootloader or ABOOT bootloader depending on the Android model. Using a reverse engineering tool, it is possible to analyze the bootloader and identify the commands used for firmware updates.

To update firmware or acquire data by accessing flash memory, the smartphone should be in the firmware update mode rather than the normal boot mode. Because only the bootloader and USB module are activated in this mode, the integrity of the acquired data is guaranteed even after physical acquisition multiple times. Thus, if the evidence phone is booted in the firmware update mode with its power off and physical acquisition is then executed, the integrity can be preserved to dump an image of the flash memory. Fig. 1 shows the displays when Samsung, LG, and Pantech smartphones are booted in the firmware update mode.

The terms used to indicate the firmware update mode and methods when entering the firmware update mode differ among manufacturers. Table 1 shows the methods for entering the firmware update mode when a smartphone boots. USB Jig is a simple circuit that gets into the firmware update mode for smartphones. Samsung and LG smartphones can be just entered into the firmware update mode according to 300 K and 910 K Ohms of resistors between pin 4 and 5 of the microUSB connector (XDA developers, 2012a, 2012b). These USB Jig cables wouldn't be used to overcome any of the existing limitations such as Original Equipment Manufacturer (OEM) unlock.

Fig. 1. Firmware update mode (Samsung, LG, and Pantech).

Analysis of LG firmware update protocols

For LG smartphones, we analyzed the firmware update processes and commands by decompiling the bootloader and the update program (LG Software & tools Download, 2014) provided by LG, and we identified the read command used for dumping the flash memory.

LG firmware update commands

LG firmware update protocol operates in the structure when sending a command packet and receiving a reply packet. The packets use the High-Level Data Link Control (HDLC) frame structure, starting with HDLC flag (0x7E) followed by packet data and Cyclic Redundancy Check (CRC)-16, and ending with HDLC flag (0x7E). Fig. 2 shows

Table 1
Methods used for entering the firmware update mode.

Model	Mode name	Key combination
Samsung Galaxy	ODIN	Press and hold, at the same time, *Volume Down*, *Home* and *Power* key (then press *Volume Up* key) or Connect 300 K ohm USB Jig
LG Optimus	DOWNLOAD	Press and hold *Volume Up* key + plug in the phone into a computer using a microUSB cable or Connect 910 K ohm USB Jig
Pantech Vega	PDL download	Press and hold, at the same time, *Volume Up*, *Volume Down*, *Home*, and *Power* key
Google Nexus 4/5	DOWNLOAD	Press and hold *Volume Up* key + plug in the phone into a computer using a microUSB cable or Connect 910 K ohm USB Jig

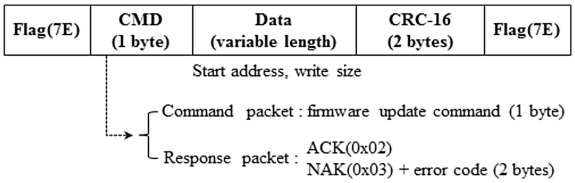


Fig. 2. LG firmware update command format.

the firmware update packet format used by LG smartphones.

After entering the LG firmware update mode, the commands for acquiring device information, GUID Partition Table (GPT) partition information, and memory information can be used. Table 2 shows the LG firmware update commands.

LG firmware update processes are as follows.

1 Get device information: 0x00.
2 Switch to download mode: 0x3A.
3 Query features (Protocol ver., etc.): 0x2F.
4 Get Partition information: 0x30.
5 Get factory information: 0xFA.
6 Write sector (Primary GPT): 0x39.
7 Repeat of partition flashing: 0x39.
8 Complete download: 0x38.
9 Reset system: 0x0A.

LG flash memory dump command

Based on the results obtained by reverse engineering the firmware update commands in the bootloader, we identified the read commands for flash memory in addition to those shown in Table 2. Fig. 3 shows the read command (0x50) for flash memory obtained by reverse engineering the SBL3 bootloader in the Optimus G model (LG-E975).

After entering the firmware update mode according to the process described in Table 1, the command for flash memory information acquisition (0x30) is sent to obtain the size of the flash memory and GPT partition information. The acquired information comprises the size of the flash memory, number of partitions, start address, end address, and name of each partition. The read command for the flash memory (0x50) is sent with the start address and dump size, and the requested size of the flash memory data can then be acquired. Fig. 4 shows the

Table 2
LG firmware update commands.

Command	Description
0x00	Get device information (model, compile date)
0x3A	Go download mode
0x2F	Get protocol version, algorithm version
0x30	Get MMC and Partition information
0xFA	Get factory information (IMEI, Mac address)
0x12	Read RAM memory
0x39	Write flash memory
0x0A	Reset system

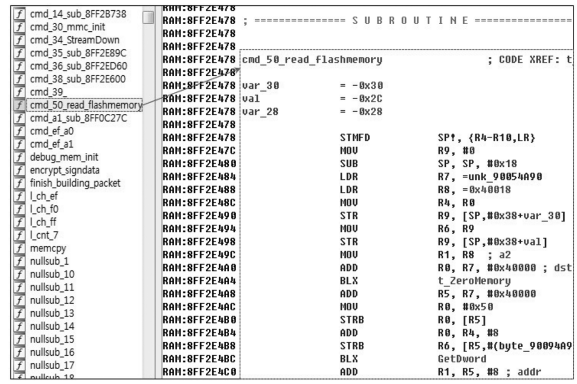


Fig. 3. Reverse engineering of the LG SBL3 bootloader.

Read command

0x50	Sub command	Start address (4 bytes)	Dump size (4 bytes)	CRC

Fig. 4. LG read command format (0x50).

format of the read command (0x50) for flash memory. Fig. 5 shows an example of data acquisition. The physical acquisition of all models can be performed using the read command regardless of Android OS and kernel version. Moreover, because it is booted in the firmware update mode, the integrity of the dump image is always maintained.

Analysis of Pantech firmware update protocol

We also analyzed the firmware update process and commands by reverse engineering the firmware update program (Pantech Self-Upgrade, 2014) provided by Pantech and the bootloader, and we identified the flash memory read command.

PC -> Phone

```
Send 0x1c bytes to the device
 7E 50 01 01 00 00 00 00 00 00 80 02 00 00 02 00   ~P........€.....
 00 00 00 00 00 00 00 00 00 00 7C 1B 7E            ..........|.~
```

Phone -> PC

```
000003: Bulk or Interrupt Transfer (UP), 26.01.2015 15:07:07.077 +0.
Pipe Handle: 0x123046b8 (Endpoint Address: 0x83)
Get 0x3f454 bytes from the device
 50 01 01 00 00 00 00 00 00 80 02 00 00 02 00 00   P........€......
 04 E7 03 00 00 00 00 00 E0 1C 41 4E 44 52 4F 49   .?.....?ANDROID!
 44 21 A0 00 74 00 00 80 20 80 63 9C 29 00 00 00   ?.t..€ €c?... ?.
 20 82 00 00 00 00 00 00 10 81 00 01 20 80 00 08   ......?.. €.....
 00 00 00 00 00 00 38 06 FA E0 89 76 6D 61 6C 6C
 6F 63 3D 36 30 30 4D 20 63 6F 6E 73 6F 6C 65 3D   0M console=ttyHS
 74 74 79 48 53 4C 30 2C 31 31 35 32 30 30 2C 6E   L0,115200,n8 lpj
 38 20 6C 70 6A 3D 36 37 36 37 37 20 75 73 65 72   =67677 user_debu
 5F 64 65 62 75 67 3D 33 31 20 6D 73 6D 5F 72 74   g=31 msm_rtb.fil
 62 2E 66 69 6C 74 65 72 3D 30 78 30 20 65 68 63   ter=0x0 ehci-hcd
 69 2D 68 63 64 2E 70 61 72 6B 3D 33 20 63 6F 72   .park=3 coresigh
 65 73 69 67 68 74 2D 65 74 6D 2E 62 6F 6F 74 5F   t-etm.boot_enabl
 65 6E 61 62 6C 65 3D 30 20 61 6E 64 72 6F 69 64   e=0 androidboot.
 62 6F 6F 74 2E 68 61 72 64 77 61 72 65 3D 67 65   hardware=geehrc8
```

Fig. 5. LG data acquisition example (LG-F240S).

Pantech firmware update commands

In Pantech models, the firmware update commands are implemented in the ABOOT bootloader. Thus, the firmware update commands used in the update process can be analyzed by reverse engineering the ABOOT bootloader.

Table 3 shows the firmware update commands used in Pantech smartphones. The firmware update commands are executed after entering the firmware update mode, as shown in Table 1. The length of the command packet varies according to the smartphone model. Thus, a 32-byte command packet was used up to the Vega Iron (IM-A870S) model, whereas a 128-byte packet was used in subsequent models.

Pantech firmware update processes are as follows.

1 Get device information: AT*PHONEINFO.
2 Switch to download mode: AT*PDL*START.
3 Start firmware update: 0x00.
4 Ready partition flashing: 0x02.
5 Erase partition with sector size: 0x04.
6 Write partition with sector size: 0x05.
7 Repeat of processes 5 and 6.
8 Complete partition flashing: 0x03.
9 Reset system: 0x01.

Pantech flash memory dump command

Based on the results obtained by reverse engineering the firmware update commands in the bootloader, the 0x06 command was found to be the read command for flash memory. Fig. 6 shows the structures of the read commands with packet lengths of 32 bytes and 128 bytes.

Information related to the GPT partition should be acquired before physical acquisition of the flash memory. The GPT partition (partition ID = 0x0A) is acquired using the read command for flash memory (0x06) and the partition information is then analyzed. Based on the results of the analysis, physical acquisition is executed for a partition or for the entire flash memory. The physical acquisition of all models can be performed regardless of the Android OS and kernel version. Because it is booted in the firmware update mode, the integrity of the dump image can always be maintained, as found in the LG models. Fig. 7 shows an example of data acquisition from the Vega Iron2 (IM-A910S) model.

Analysis of Samsung firmware update protocol

For Samsung smartphones, we analyzed the firmware update protocol in the bootloader and found that there were the read commands for the flash memory but the actual code for reading the flash memory had been removed.

Table 3
Pantech firmware update commands.

Command	Description
0x00	Firmware update ready command
0x01	Reset command
0x02	Write ready command
0x03	Write complete command
0x04	Partition erase command
0x05	Partition write command

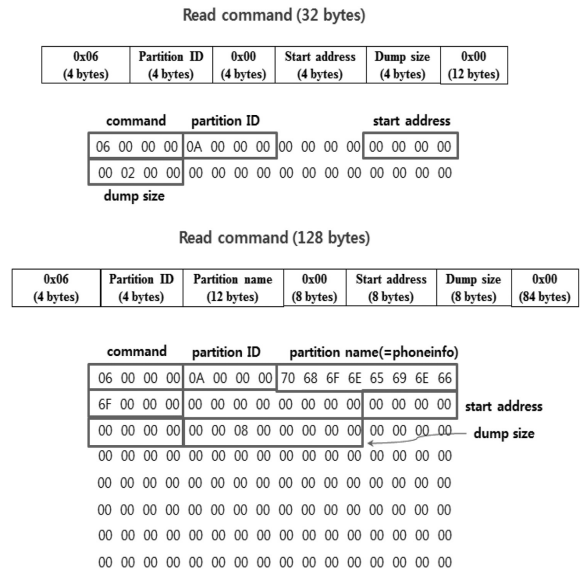


Fig. 6. Pantech read command for flash memory (0x06).

Samsung firmware update commands

In the Samsung firmware update protocol, a PC sends a command packet, and a smartphone processes the command and sends back the results in a reply packet. The command packet comprises 1024 bytes, and it executes a different function according to the values of the first 4 bytes. Fig. 8 shows the format of the Samsung firmware update command. Table 4 shows the firmware update commands used in Samsung models.

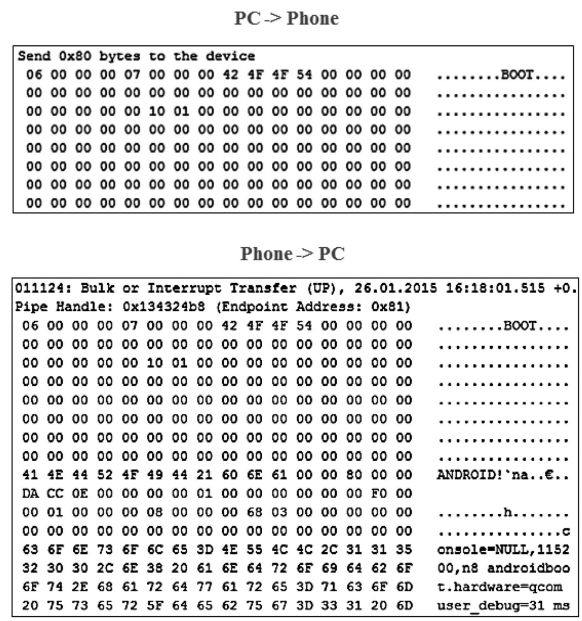


Fig. 7. Pantech data acquisition example (IM-A910S).

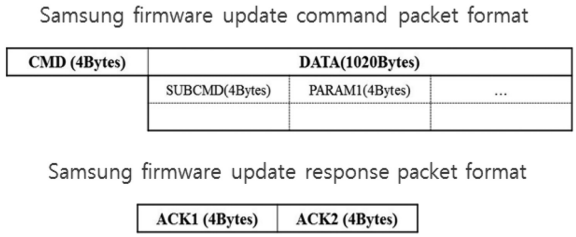


Fig. 8. Samsung firmware update command format.

The firmware update processes are as follows. The firmware update protocol is initialized by sending the 0x64 command packet. After sending the 0x65 command packet, the partition information table of the new firmware is then uploaded in the main memory. Next, the 0x66 command packet is sent and the firmware based on the new partition information is flashed in the flash memory, and the 0x67 command packet is set to complete the firmware update process. The 0x66 sub commands (0x01, and 0x03) can acquire data from the flash memory, but the actual dump code has been removed. If the read command (CMD: 0x66, SUBCMD: 0x01, 0x03) is sent, the smartphone only sends an ACK message without data. We only checked whether the flash memory read function had been removed in the Galaxy S2 and subsequent models. Fig. 9 shows the read command by reverse engineering Samsung bootloader using the IDA Pro tool. It is difficult to acquire data of flash memory using the read command because the dump code is removed. Thus, in order to conduct physical acquisition, additional research for patching the read command is required.

Android physical acquisition process

After analyzing the firmware update protocols of LG and Pantech smartphones, we found that the read command for flash memory was preserved and physical acquisition could

Table 4
Samsung firmware update commands.

Command	Sub commands	Description	
0x64	0x00	Firmware update protocol initialization command	–
0x65	0x00	Partition information table write ready command	–
	0x01	Partition information table read ready command	–
	0x02	Partition information table write/read command	–
0x66	0x00	Write ready command for flash memory	–
	0x01	Read ready command for flash memory	Code removal
	0x02	Write command for flash memory	–
	0x03	Read command for flash memory	Code removal
0x67	0x01	Firmware update complete command	–

```
int __fastcall process_packet_102_update_firmware(int result)
{
  int v_cmd_buf; // [sp+4h] [bp-18h]@1
  int v2; // [sp+8h] [bp-14h]@3
  int v_cmd; // [sp+Ch] [bp-10h]@1
  int v_binary_phone; // [sp+10h] [bp-Ch]@10
  int v_status; // [sp+14h] [bp-8h]@13

  v_cmd_buf = result;
  v_cmd = (*(_BYTE *)(result + 3) << 24) | (*(_BYTE *)(result + 2) << 16)
  switch ( (*(_BYTE *)(result + 7) << 24) | (*(_BYTE *)(result + 6) << 16)
  {
    case 0:
      id_102_flag = 0;
      result = upload_ack(v_cmd, 0);
      break;
    case 1:
      id_102_flag = 1;
      result = upload_ack(v_cmd, v2);
      break;
    case 2:
      if ( id_102_flag != 1 && !id_102_flag )
      {
        v2 = (*(_BYTE *)(result + 11) << 24) | (*(_BYTE *)(result + 10) <<
        upload_ack(v_cmd, 0);
        result = download_data(v2);
      }
      break;
    case 3:
      if ( id_102_flag == 1 )
      {
        result = upload_ack(v_cmd, 0);
      }
      else if ( !id_102_flag )
      {
        v_binary_phone = (*(_BYTE *)(result + 11) << 24) | (*(_BYTE *)(res
        if ( !dit.set_nps_update )
```

Code removal (CMD: 0x66, SUBCMD: 0x01)

Code removal (CMD: 0x66, SUBCMD: 0x03)

Fig. 9. Reverse engineering of the read command in bootloader.

be performed using this command. There was the flash memory read command in the firmware update commands in Samsung models, but the actual dump code had been removed. Based on this analysis, we developed a new physical acquisition procedure for Android smartphones. First, check whether there is the flash memory read command by reverse engineering the firmware update protocol in the bootloader. If there is a read command, execute physical acquisition in the LG and Pantech models. If not, additional research for patching the read command is required.

Android physical dump

Using the acquisition method proposed in the paper, we developed an acquisition tool called Android Physical Dump (APD) and it is written in C++. Fig. 10 shows the APD tool. After connecting via a USB cable, it selects buttons for physical acquisition.

Supported models

The APD tool currently supports over 80 of the latest Android models. Currently, APD can perform physical dumps from LG Optimus, Pantech Vega, and Google Nexus models. Representative examples of the models supported are as follows: LG G3, G2, G, Pantech R3, Iron2, Nexus 4/5, and G Watch.

Boot with firmware update mode

In the first physical dump step, the smartphone is booted in the firmware update mode, as described in Table

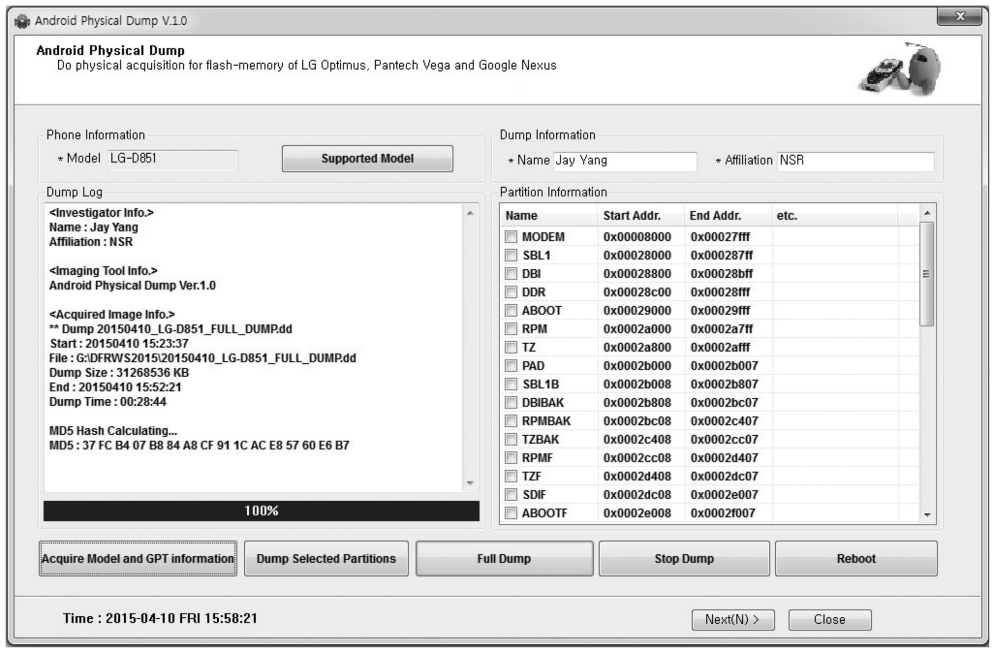


Fig. 10. Android physical dump (APD).

1. Smartphones can enter the firmware update mode via the AT command or the download mode command. However, if physical memory acquisition is executed after normal boot, the hash value of the acquired data can be changed each time that acquisition is performed. Therefore, after the evidence phone has been collected, it needs to be turned off and booted with the firmware update mode.

Connect to the phone and acquire the model information

Connect to the smartphone using a USB cable. The USB driver must be installed in advance. The USB driver can be downloaded from the manufacturer's homepage. After connecting a USB cable, select the Acquire Model and GPT information button. If the user only selects the manufacturer, the smartphone model name is displayed automatically. After acquiring the model information, the partition information is obtained by sending the GPT partition information command. As shown in Fig. 10, the window presents the partition name, start address, and end address for each partition.

Physical acquisition

The APD tool was implemented to support a partition dump and a whole flash memory dump. After the Dump Selected Partitions button is selected, a physical dump of the relevant partition is performed. After the dump process, the dump time and MD5 hash value of the dump image are displayed in the Dump Log window.

The Full Dump button is selected to acquire the entire flash memory. The physical acquisition process is executed using the start and end addresses of the flash memory in the acquired GPT partition information. After the acquisition process is complete, the investigator information, acquisition tool information, and dump information are displayed, as shown in Fig. 10. The dump information indicates the starting time and ending time, where the MD5 hash value is calculated and displayed for the dump image. Hash value calculation is important for checking the integrity of the dump image.

The file acquired using the APD tool is raw data format and can be analyzed through smartphone forensic tools (Cellebrite UFED Physical Analyzer, 2015; Guidance EnCase, 2014; R-Linux, 2014).

Experiments

The most important factors in the area of Android forensic acquisition are guaranteeing the integrity of the dump image, rapid evidence collection, and dump availability in an anti-forensic environment due to the pattern lock and user password. Based on these three factors, we compared the APD tool proposed in the present study with existing acquisition methods using the latest Android smartphones. The tools compared were the well-known Cellebrite UFED 4PC, dump method based on a custom recovery image, ADB physical dump via rooting exploitations, and JTAG-based acquisition. Table 5 shows the experimental results obtained. The following different models were used in the tests: G3 (F400S, D851), Optimus G (F180S, E975), R3 (IM-A850S), Iron2 (IM-A910S), and Nexus 4/5 (E960, D821). After completing the process of dumping the flash memory with the APD tool, we would check the dumped image using the Cellebrite UFED Physical Analyzer (2015) to determine the quality of the acquired dumped image.

Table 5
Experimental results.

Item	Proposed method (APD)	Cellebrite UFED 4PC	Custom recovery mode dump	ADB dump using root exploitation	JTAG-based acquisition
Integrity guaranteed	O	X	X	X	O
Acquisition speed (size: 32 GB)	30 min	120 min	120 min	180 min	480 min[a]
Dump of smartphone with screen-lock	O	X	X	X	O

[a] JTAG-based acquisition method excludes the disintegration and connection time.

Preserving the integrity of the dumped image

In a previous study (Son et al., 2013), the user data integrity was checked with a JTAG-based acquisition method. This method guarantees the integrity of the user data because it only flashes a custom recovery image. However, the integrity of the entire flash memory is damaged because the recovery partition of the flash memory is modified. The integrity is also damaged in Cellebrite UFED 4PC and ADB dump using rooting exploitation because the acquisition process is executed after the normal boot.

By contrast, the proposed acquisition method preserves the integrity of the entire flash memory. It boots in the firmware update mode and physical acquisition is performed using the flash memory read command. Thus, we compared the JTAG-based acquisition method to confirm whether the integrity of the entire flash memory was maintained. For the accuracy of the results, the experimentation processes were followed as the previous study (Son et al., 2013). We compared the hash values of the dumped images after acquiring five times.

Acquisition speed

Due to the rapid increase in the use of smartphones, the number of smartphones that needs to be analyzed is also increasing sharply. Because the number of evidence phones that need to be examined is large, it is not easy to use the JTAG-based acquisition method, which requires an acquisition time that exceeds 8 h in the field. Thus, fast S/W-based acquisition methods are often used. Therefore, we conducted an experiment to compare the acquisition time for the entire flash memory and the results are shown in Table 5. The results represent the mean values of the acquisition times of 8 models. The proposed method can set the maximum size of the smartphone to send data to a PC, so it required about 30 min on average to acquire the 32 GB flash memory, which was about four times faster than the acquisition time with the other methods.

Physical acquisition from screen-locked smartphones (USB debugging disabled)

Most S/W-based forensic tools use the ADB protocol for physical acquisition. USB debugging must be enabled on the smartphone to use the ADB protocol. However, all Android smartphones are delivered with USB debugging disabled for security reasons. Thus, USB debugging should be enabled in order to apply existing acquisition methods. Therefore, it is impossible to execute physical acquisition using one of the existing methods in a smartphone that is locked by a pattern or user password (USB debugging disabled). This shortcoming is an important issue that needs to be resolved in the area of Android physical acquisition. However, the proposed method overcomes this problem. Even for a screen-locked smartphone, it is possible to execute physical acquisition after turning off the phone and rebooting in the firmware update mode.

Conclusion

We developed a new method for acquiring the entire flash memory by analyzing the firmware update protocols of Android smartphones. We decompiled the firmware update programs provided by manufacturers and analyzed the firmware update protocols in the bootloader. Based on the analytical results of firmware update protocols, we found that the flash memory read commands were preserved in some firmware update protocols and thus physical acquisition could be executed. When the flash memory read commands have been removed, additional research for patching the read command is required. Based on this new acquisition method, we developed an acquisition program to support the physical dumping of flash memory for over 80 of the latest Android models.

By comparing the proposed tool with existing methods, we proved that our method guarantees the integrity of the entire flash memory, which it acquires at a high speed, and physical acquisition can be executed regardless of the restriction due to screen locking with a pattern or user password (USB debugging disabled).

The proposed acquisition method has the limitation that it is necessary to analyze the firmware update protocol whenever new Android smartphones are launched. However, because the firmware update protocol is implemented in the bootloader of all Android smartphones and each manufacturer applies the same firmware update protocol to all of its models, physical acquisition can be performed for all models of the manufacturers as long as the acquisition method by analyzing the firmware update protocol is found. Thus, continuous research is required in this area.

References

AccessData MPE+ http://accessdata.com/solutions/digitalforensics/mpe; 2014.
Android Backup Extractor. http://sourceforge.net/projects/adbextractor/; 2014.
Android Debug Bridge (ADB). http://developer.android.com/tools/help/adb.html.
Android software development – fastboot. http://en.wikipedia.org/wiki/Android_software_development#Fastboot; 2014.

Breeuwsma M, Jongh M, Klaver C, Knijff R, Roeloffs M. Forensic data recovery from flash memory. Small Scale Digital Forensics J 2007;1(1):1–17.
Bring your own device. http://en.wikipedia.org/wiki/Bring_your_ own_ device; 2014.
Cellebrite UFED 4PC. http://www.cellebrite.com/Mobile-Foensics/Products/ufed-4pc; 2015.
Cellebrite UFED Physical Analyzer. http://www.cellebrite.com/Mobile-Forensics/Applications/ufed-physical-analyzer; 2015.
Grover J. Android forensics: automated data collection and reporting from a mobile device. Digit Investig 2013;10:S12–20.
Guidance EnCase. http://www.guidancesoftware.com/; 2014.
Hannay P. Kindle forensics: acquisition & analysis. Proc Conf Digital Forensics, Secur Law 2011;6(2):143–50.
Hex-Rays. https://www.hex-rays.com/index.shtml/; 2015.
High-Level Data Link Control (HDLC). http://en.wikipedia.org/wi ki/High-Level_Data_Link_Control.
Hoog A. Android forensics. Mobile forensics world 2009.
Hoog A. Android forensics: investigation, analysis and mobile security for Google Android. Syngress; 2011.
HTC Sync Manager. http://www.htc.com/us/software/htc-sync-manager/; 2014.
Iqbal A, Alobaidli H, Baggili I, Marrington A. Amazon kindle fire HD forensics. In: Digital forensics and cyber crime; 2013. p. 39–50.
Jovanovic Z. Android forensics techniques. International Academy of Design and Technology; 2012.
Kim K, Hong D, Ryu J. Forensic data acquisition from cell phones using JTAG interface. Information Security Research Division; 2008. p. 410–4.
Lessard J, Kessler G. Android forensics: simplifying cell phone examinations. Small Scale Digital Device Forensics J 2010;4(1):1–12.
LG Software & tools Download. http://www.mylgphones.com/lg-software-tools-download; 2014.
MSAB XRY. https://www.msab.com/products/xry; 2015.
MultiMediaCard. http://en.wikipedia.org/wiki/MultiMediaCard #eMMC; 2014.
Mutawa N, Baggili I, Marrington A. Forensic analysis of social networking applications on mobile devices. Digit Investig 2012;9:S24–33.
ORT tool. http://www.orttool.com; 2014.
Oxygen Forensics. http://www.oxygen-forensic.com/; 2014.
Pantech Self-Upgrade. http://www.pantechservice.co.kr/down/self/main.sky; 2014.
R-Linux. http://www.r-tt.com/free_linux_recovery/index.shtml/; 2014.
RIFF box. http://www.riffbox.org; 2014.
Rooting (Android OS). http://en.wikipedia.org/wiki/Rooting_ (Android_ OS); 2014.
Samsung Kies. http://www.samsung.com/us/kies/; 2014.
Samsung KNOX. http://www.samsungknox.com/; 2014.
Samsung Odin. http://odindownload.com/; 2014.
Secure Boot. https://source.android.com/devices/tech/security/secureboot/index.html; 2014.
Smartphone OS Market Share Q3 2014. http://www.idc.com/prodserv/smartphone-os-market-share.jsp; 2014.
Son N, Lee Y, Kim D, James J, Lee S, Lee K. A study of user data integrity during acquisition of Android devices. Digit Investig 2013;10:S3–11.
Sony PC Companion. http://support.sonymobile.com/gb/tools/pc-companion/; 2014.
Sylve J, Case A, Marziale L, Richard G. Acquisition and analysis of volatile memory from android devices. Digit Investig 2012;8:175–84.
Vidas T, Zhang C, Christin N. Toward a general collection methodology for Android devices. Digit Investig 2011;8:S14–24.
XDA developers. How to make your own usb jig. 2012. http://forum.xda-developers.com/galaxy-s2/help/guide-how-to-make-use-jig-reset-binary-t1604707.
XDA developers. LG 910K USB flash cable scheme. 2012. http://forum.xda-developers.com/showthread.php?t=2069564.
Xiaomi Download MiFlash for Xiaomi Smartphone. http://www.jayceooi.com/download-miflash-for-xiaomi-smartphone/; 2015.
Z3X box. http://z3x-team.com; 2014.