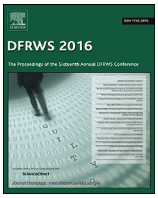
Digital Investigation 18 (2016) S46–S54

Contents lists available at ScienceDirect

Digital Investigation

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



DFRWS USA 2016 — Proceedings of the 16th annual USA digital forensics research conference

Rapid differential forensic imaging of mobile devices

M. Guido[*], J. Buttner, J. Grover

The MITRE Corporation, 7515 Colshire Drive, Mclean, VA, 22102, USA

Keywords:
Mobile forensics
Rapid acquisition
Differential analysis
Android

ABSTRACT

Commercial mobile forensic vendors continue to use and rely upon outdated physical acquisition techniques in their products. As new mobile devices are introduced and storage capacities trend upward, so will the time it takes to perform physical forensic acquisitions, especially when performed over limited bandwidth means such as Universal Serial Bus (USB). We introduce an automated differential forensic acquisition technique and algorithm that uses baseline datasets and hash comparisons to limit the amount of data sent from a mobile device to an acquisition endpoint. We were able to produce forensically validated bit-for-bit copies of device storage in significantly reduced amounts of time compared to commonly available techniques. For example, using our technique, we successfully achieved an average imaging rate of under 7 min per device for a corpus of actively used, real-world 16 GB Samsung Galaxy S3 smartphones. Current commercially available mobile forensic kits would typically take between one to 3 h to yield the same result. Details of our differential forensic imaging technique, algorithm, testing procedures, and results are documented herein.

Introduction

Performing a physical forensic acquisition for Android™ devices usually requires the device to be booted into one of the following environments:

- Custom bootloader
- Custom recovery mode
- Normal mode with root access.

In any of these modes, physical acquisition techniques need to execute code on a target mobile device, which creates in-memory, bit-for-bit copies of the device that are sent to a receiving service. In most commercial toolkits, data is typically sent over a Universal Serial Bus (USB) interface to a connected hardware device or server (e.g., laptop or desktop). USB 2.0 has a maximum transmission rate of 480 Mb/s but rarely achieves speeds of more than 320 Mb/s (Spector, 2014).

Full physical forensic acquisitions can be time-intensive, sometimes taking hours to complete. Acquisition times are largely dependent on device processor speeds, cable types used, and the amount of data transferred. At the time of writing, the largest Android devices available on the market were 128 GB (Florin, 2015). As devices continue to grow in size, the times to physically acquire them will likely increase.

In some circumstances, such as time-sensitive operations at crime scenes or border crossings, having a rapid physical acquisition capability for mobile devices could be critical in resolving a situation. Currently, in these situations, a forensic investigator may instead opt to perform a logical device acquisition to save time. Not having a physical image available during an examination may open questions about missing data, as logical acquisitions do not capture disarranged or deleted files, and in some cases may not preserve file timestamps.

Our research focuses on reducing the amount of data that needs to be transferred during the physical acquisition

[*] Corresponding author. Tel.: +1 703 983 5130; fax: +1 703 983 1002.
E-mail address: mguido@mitre.org (M. Guido).

http://dx.doi.org/10.1016/j.diin.2016.04.012
1742-2876/

process, thus decreasing the overall acquisition time. The final product can be the same as that of a traditional acquisition tool: a complete physical forensic image. We utilized prior research to develop a prototype software agent named *hawkeye*, which uses differential analysis and runs within an Android custom bootloader or recovery mode to acquire a physical forensic image.

The remainder of this paper is structured as follows: Section (Related work) covers related work; Section (Corpus of phone images) discusses the phone image corpus used by *hawkeye*; Section (Hawkeye) contains *hawkeye* implementation details, including the algorithm used; Section (Experimentation) contains experiment procedures and results; Section (Discussion) includes a discussion area; and Section (Conclusion) concludes with a summary and some proposed future work.

Related work

The Hawkeye project is an extension of the Periodic Mobile Forensics (PMF) system (Guido et al., 2013); however, it targets a different use case (e.g., a crime scene) and is specifically designed to improve device acquisition speeds. The *hawkeye* agent is designed to operate as a client within the overall PMF system architecture, which is referenced heavily within this work. Both systems use differential forensic analysis, as formally defined by Garfinkel, Nelson, and Young (Garfinkel et al., 2012). The original PMF agent has operated on a variety of mobile devices running Android 2.2+, and there is no reason foreseen that the *hawkeye* agent, using the lessons learned building PMF compatibility, should not be considered equally compatible with modern mobile devices.

Laurenson et al. applied and automated the work done by Garfinkel, Nelson, and Young to collect and distribute application software artifacts in a reference set that they termed *application profiles* (Laurenson et al., 2015). While their purpose and implementation differ, there are many similarities found in building Hawkeye's baseline hash list and corresponding data storage in PMF. We will describe several mechanisms built into PMF (Guido et al., 2013) for generating these hashes.

Gurjar et al. (2015) compared the runtime efficiency of common hashing algorithms (MD5, SHA-family) and their implementations on Windows® and Linux®. In their work, they found that MD5 performed best on both Windows and Linux. Hawkeye uses MD5 as its hash algorithm because of its speed. The risk of constructed collisions using MD5 is not relevant for the scope of our work.

The method of using hash maps to discriminate known-good files is well known in the forensics community, although it is not typically implemented in physical acquisition tools. This fact holds true in the commercial mobile forensic acquisition tools that were tested as part of our work. One of the notable contributions of the Hawkeye work is the implementation of differential analysis, enabling Hawkeye to only hash a fraction of a device's storage, leading to significant time savings.

Watkins et al. (2009) previously developed Teleporter. The *hawkeye* agent incorporates a hashing comparison technique similar to that of Teleporter; however, both the purpose and overall system design differ significantly. Teleporter's purpose is to enable transport of a minimal amount of data from hard drives when faced with limited bandwidth environments, sometimes over large distances, and it was designed to identify both known files and previously recorded blocks of data. Hawkeye's purpose is to acquire a full disk image from a target mobile device and does not have requirements to interpret any filesystems or storage content. Some partitions of an Android device are structured in proprietary or undocumented formats; Hawkeye acquires them all and makes no attempt to understand them; that is left for the PMF system (Guido et al., 2013). Similarly, Grier and Richard (2015) use sifting collectors to identify and acquire only the regions of a disk that have forensic value. Their research limits the amount of device data imaged and does not result in complete bit for bit copies produced by Hawkeye or other mobile forensic tools. Grier and Richard's approach is similar to Teleporter in that they interpret filesystems to identify files and they note that their methods are not suitable for unknown filesystems.

Garfinkel et al. (2010) performed forensic analysis at the block and sector level. They developed algorithms to identify fragments of file formats on a storage device and showed that contents of a storage device can be determined with high accuracy using statistical sampling. Hawkeye uses hash representations of much larger blocks of storage compared to (Garfinkel et al., 2010), primarily to tradeoff the number of required hash comparisons performed to the amount of data transmitted over the wire. Statistical sampling to determine mobile storage contents could be complementary during forensic analysis of the mobile device images that Hawkeye collects.

Mobile forensic acquisitions are often considered “live acquisitions” because they rely on a target device's running kernel. Many commercially available mobile phone kits use live acquisition techniques to take one-time logical or physical images of a target device (Lessard and Kessler, 2010). Vidas, Zhang, and Christin (2011) developed a more generalized acquisition method that requires no prior knowledge of phone content. Son et al. (2013) studied the recovery mode method formally introduced by Vidas, Zhang, and Christin and found that a device's userdata partition maintained its integrity during the “recovery mode” acquisition. Recovery mode is a preferred environment for *hawkeye* to execute in because it enables the tool to operate on many different Android devices, provides access to Android API functions, and temporarily disables all wireless functionality of the mobile device.

Yang et al. (2015) demonstrated a new method of acquiring a device through the Android update protocols of some devices' bootloaders. They tested a variety of 32 GB Android devices[1] and found that their method was significantly faster than the Cellebrite® UFED 4PC. They stated that their method took 30 min and UFED 4PC took 120 min on average. Their results inspired the optimization and

[1] Devices used by Yang et al.: LG® G3™ (F400S, D851), Optimus G™ (F180S, E975), R3 (IM-A850S), Iron2 (IM-A910S), and Nexus™ 4/5 (E960, D821).