

Validation of forensic images for assurance of digital evidence integrity

Honours Thesis

James Michael McCutcheon

School of Engineering and Information Technology

Murdoch University

November, 2014

Abstract

The reliability of digital evidence is an important consideration in legal cases requiring sound validation. To ensure its reliability, digital evidence requires the adoption of reliable processes for the acquisition, preservation, and analysis of digital data. To undertake these tasks, the courts expect digital forensic practitioners to possess specialised skills, experience, and use sound forensic tools and processes. The courts require that the reliability of digital evidence can be verified with supporting documentation; notably acquisition process logs and a chain of custody register, confirming that the process of recovering and protecting the evidence was based on sound scientific principles.

In typical cases the digital evidence has been 'preserved' in a special file or 'container' that has been declared to be secure on the basis that it is not possible to tamper with the contents of the container or the information supporting the contents (metadata) without this act being discovered. However, through the use of a freely available open source library, libewf, it has been discovered that the most commonly used forensic container format, Encase Evidence File Format, also known by its file extension .E01, can be manipulated to circumvent validation by forensic tools. This digital forensic container contains an embedded forensic image of the acquired device and metadata fields containing information about the data that was acquired, the circumstances of the acquisition, and details about the device from which the forensic image was acquired. It has been found that both the forensic image and the metadata associated with that image can be freely altered using simple file editors and open source software.

Exploiting these weaknesses within the Encase Evidence File format results in a forensic container that can be altered but fails to provide any evidence that this has occurred. In practice the original device is often unavailable, damaged, or otherwise unable to provide independent validation of the data held in the container. In such situations, it would be difficult, if not impossible, to determine which of two forensic containers held the original record of the evidence.

As part of a proof of concept, existing libewf code was manipulated to allow for legitimate metadata to be attached to a compromised and altered forensic image with recalculated hashes and data integrity checksums. Without incontrovertible records of the original data's hash value, this manipulation might only be detected by an independent third party holding a copy of the original forensic container's metadata and hashes for comparison. While hashes and metadata held by an interested party could also potentially be altered or declared unreliable, an uninterested party would be able to provide a more reliable set of hashes that could be used to validate the unaltered container.

In order to add to the body of knowledge supporting digital forensics as a scientific discipline this research has brought into question a fundamental assumption about the reliability of a fundamental method currently used to collect and validate digital evidence. Further research is required to determine the whether processes can be designed to enhance the detection of contaminated images.

Acknowledgements

I would like to acknowledge the immense guidance and support provided by my supervisors, Dr Richard Adams and Mr Richard Boddington. Their professional and academic advice and experience has been extremely helpful in shaping this thesis and their enthusiasm for and belief in the importance of my research encouraging and empowering. Thank you both for your valuable time and effort over the past months.

I would also like to acknowledge Mr Matthew Davies for his advice and guidance towards understanding the libewf code base and resolving the technical challenges related to the proof of concept code developed as part of the research.

Contents

1	Introduction	1
2	Literature review	3
2.1	Digital forensics	3
2.1.1	The purpose of digital forensics	4
2.1.2	The nature of digital evidence	6
2.2	Digital forensic processes	8
2.3	Preservation and recovery of digital evidence	11
2.3.1	The role and importance of crime scene preservation	11
2.3.1.1	The acquisition process	13
2.3.1.2	Chain of custody	15
2.3.2	Sources of contamination and preservation issues	17
2.3.2.1	BIOS addressing limitation	18
2.3.2.2	Bad sectors	18
2.3.2.3	Hardware level data obfuscation	19
2.3.3	The importance of forensic image validation and preventing contamination	19
2.3.3.1	The role and importance of hashing and digitally signing the forensic image	21
2.3.3.2	Tools available to prevent contamination	24
2.3.3.3	Forensic containers	25
2.4	Research Directions	26
2.5	Summary	28
3	Research objectives	29
3.1	Digital evidence container metadata	29
3.2	Research questions and hypotheses	30
3.2.1	Alteration of digital evidence container metadata	31
3.2.2	Alteration of forensic images within digital evidence containers	31
3.2.3	Detection of alterations to digital evidence containers	32
3.2.4	Differentiation of forensic image containers	32
4	Experiment design and results	34
4.1	Equipment	35
4.2	Testing environment	35
4.3	Testing of H ₁	36
4.4	Testing of H ₂	39
4.5	Testing of H ³	43
4.6	Testing of H ₄	45
4.7	Summary of results	48
5	Discussion of findings	49

5.1	Altering metadata	49
5.2	Altering forensic images	50
5.3	Recommendations	50
5.4	Further research	52
6	Conclusion	54
	References	55
	Appendices	61
	Appendix A	61
	Appendix B	66

1 Introduction

Cyber transgressions, whether criminal acts, civil wrongs, or personnel misconduct, are on the increase due to an increasing access to and reliance upon technology in everyday life (Nfuka, Sanga, & Mshangi, 2014). These transgressions require investigation but often involve technologies that exceed the skills and experience of investigators and legal practitioners, thus creating a need for specialist digital forensic practitioners (Rogers, Scarborough, Frakes, & San Martin, 2007). This thesis focuses on a key aspect of digital forensics: the validation of digital evidence recovered from electronic devices.

The chain of custody is a record of the handling and protection processes used to ensure the continuing integrity of physical evidence seized from a crime scene, including computing devices holding digital evidence, such as computer hard drives and mobile phones (Cosic & Baca, 2010). The chain of custody documents the state, location, and interactions with the evidence and is intended to confirm the authenticity of the evidence between seizure and presentation in legal hearings (Jarrett, Bailie, Hagen, & Judish, 2009, p.197).

The evidence acquisition process logs created by digital forensic acquisition software are a record of the handling, extraction, and copying processes used to ensure the integrity of the digital evidence located on physical devices that have been seized from a crime scene (Richard, Roussev, & Marziale, 2007). These logs include information such as the date of acquisition, the amount of data acquired, hash values and other relevant details about the seized device. They are intended to demonstrate that the acquisition process created an authentic image of the digital information held on the device (CDESWFG, 2006). The evidence extracted from devices is subject to intense scrutiny in courts, and the chain of custody register and acquisition process logs assist in persuading the court of its accuracy and authenticity and hence its reliability (Jarrett et al., 2009).

This thesis explores the processes currently used to validate digital evidence intended for use in legal hearings and seeks to identify vulnerabilities in these processes that allow for the malevolent alteration of digital evidence, such that the altered evidence could not be differentiated from the original evidence when supporting documentation is considered unreliable or is was unavailable. This research stems from the suggestions of other researchers, such as Mocas (2004) and Garfinkel (2010), that further research into the areas of data integrity and authentication processes is required.

The research undertaken seeks to test the most common processes and tools used by digital forensics investigators by attempting to take advantage the technical limitations of digital evidence containers and the software that creates them. This was conducted through a series of experiments testing the degree to which digital evidence containers can be altered and the ability of forensic software tools to detect these alterations.

This thesis consists of the following chapters: a literature review detailing digital forensic processes and previous research into the validation of digital evidence; an overview and justification of the research objectives of this project; a description of the experiments used to validate identified weaknesses in the Encase Evidence File digital evidence container; a presentation of the results of those experiments; a discussion of the findings and their implications; and a list of recommendations that can lessen the potential impact or prevent the exploitation of the weaknesses identified in the research presented in this thesis.

2 Literature review

Digital forensics is a broad discipline ranging from technical detail on the operation of electronic devices through to the legal theory applicable to digital evidence, as demonstrated by the variety of research discussed in this literature review. This literature review seeks to provide an introduction as to how technical expertise, specialised hardware and software tools, and standardised processes work together to provide reliable digital evidence in support of legal argument. The literature review provides an overview of the purpose and nature of digital forensics and provides a description of digital evidence. The acquisition of digital evidence and the evidentiary requirements it must meet are described and the review concludes with an appraisal of the condition of digital forensics research regarding digital evidence validation.

2.1 Digital forensics

Digital forensics, sometimes called cyber forensics, computer forensics, or forensic computing, is the investigation of circumstances in which a computer is believed to have been used to commit criminal, civil, or staff misconduct (Carrier, 2005; Rowlingson, 2004). The growth of crimes involving computers has resulted in an increasing reliance upon digital evidence in legal cases, including those where the evidence is collected entirely in digital form (Cohen, 2008; De Maio, 2013; Mouhtaropoulos, Li, & Grobler, 2014, p.5). Digital evidence is generally held to be information stored in a binary form (encoded as strings of 1s and 0s), in a wide range of devices, such as hard drives and mobile phones, which holds investigative or probative value to digital forensics practitioners, investigators, and legal practitioners (Ashcroft, 2001, pp.9-22; Pollitt, 2001).

The first Digital Forensics Research Workshop (p.16, 2001) agreed upon the following definition of digital forensics:

The use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of

facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations.

Digital forensics is, therefore, defined as a science using repeatable process and logical deduction to identify, extract and preserve digital evidence, a definition supported by Noblett, Pollit and Presley (2000). Altheide and Carvey (2011, p.2) assert that investigation, a part of the process, is also an art, relying on experience and intuition to analyse the available data.

2.1.1 The purpose of digital forensics

The Digital Forensics Research Workshop (2001) further categorised the aims of digital forensics as law enforcement, information warfare, and critical infrastructure protection (business and industry); while the primary goal of law enforcement is to prosecute those who break the law, prosecution may only be a secondary objective of the latter categories, if it is even considered at all.

The goal of any digital forensic examination is to identify the facts relating to an alleged event and to create a timeline of these events that represents the truth (Altheide & Carvey, 2011). A practitioner should strive to link these events to the identity of an individual; however, this may not always be possible and the events may be described with unknown actors (Digital Forensics Research Workshop, 2001).

Admissibility in legal proceedings is the appropriateness of evidence to be used as a basis for legal argument; admissible evidence must be reliable and relevant to the case (Gottesman, 1994). Casey (2011, pp. 56-57) states that determining admissibility varies between jurisdictions, though evidence must comply with all of the following conditions in that it is:

- Relevant,
- Authentic,
- Not hearsay (some hearsay is exempt),
- The best evidence available, and

- Not unduly prejudicial.

Evidence which is found to be irrelevant to the case or deemed to be unreliable, in other words unable to present objective truth, is considered inadmissible and cannot be considered in judgment of the case (Gottesman, 1994). 'Scientific' expert testimony, such as that presented by a digital forensics practitioner in United States jurisdictions, is typically assessed by the Daubert Test, from the case *Daubert v. Merrell Dow Pharmaceuticals Inc.* (1993).

The Daubert test is used in the United States as a benchmark for evaluating the validity of tools and processes used to acquire, preserve, and analyse digital evidence (Craig, Swauger, Marberry, & Hendricks, 2006). While this is not an Australian case, it was considered by the High Court of Australia in *Osland vs The Queen* (1998). A witness asserted that scientific evidence presented during the case would not have satisfied the Daubert test and the Justices agreed, noting that it also failed to meet Australian evidence standards (High Court of Australia, 2000).

The Daubert test is a set of non-prescriptive, flexible guidelines to be used by a judge to determine whether evidence presented by an expert witness can be considered to be based on scientific knowledge (Gottesman, 1994). Suggested guidelines include peer review, known error rates, empirical testing, and the adherences to standards and protocols developed and maintained by scientific peers (Gottesman, 1994). In determining the scientific credentials of the witness and their testimony the judge acts in an independent 'gatekeeper' role who assesses each case according to its individual circumstances (Gottesman, 1994).

Digital evidence, if not handled in strict accordance with protective handling and investigation guidelines, is susceptible to alteration, destruction, or other contamination that can impair its admissibility and evidentiary weight (Ashcroft, 2001; Carrier & Spafford, 2003). The tools and techniques used in digital forensics are designed to assist the practitioner in preserving and protecting digital evidence (Lenstra & de Wegner, 2005; Schneier, 2004).

Digital evidence, like other forms of indirect evidence, is provided to the legal counsel to determine its admissibility and its suitability for use in legal proceedings, based upon the rules of the court and its ability to support a convincing argument (Ashley & Rissland, 1986; Perelman, Weaver, Wilkinson, & Olbrechts-Tyteca, 1971). The legal counsel can use evidence they deem suitable to construct an argument by linking discrete pieces of evidence together using logical inference; these inferences are used to persuade the court towards a particular interpretation of the evidence presented (Silverstone & Sheetz, 2007).

2.1.2 The nature of digital evidence

Digital evidence is information that is stored in electronic devices. Carrier & Spafford (2003) categorise the storage device as physical evidence, together with any other hardware and peripherals; while binary data stored on these devices is categorised as digital evidence. Digital evidence can be recovered from a wide range of devices and take various forms, such as system/server logs, network logs, network traffic, and file system data (Sommer, 1998). File system data can be found in a variety of devices such as removable disks, both optical (CD) and magnetic, floppy disks (Duerr, Beser, & Staisiunas, 2004), magnetic tapes (Thompson & Berwick, 1998, p.36), memory cards, digital cameras, personal electronic devices, hard drives (Ashcroft, 2001, pp.9-22), and more recently flash memory and solid state drives (Bell & Boddington, 2010). The random access memory (RAM) of a computer also contains data that can be captured and analysed, provided the suspect machine has not been powered off before acquisition as RAM needs to be powered to retain the data it contains (Ruff, 2008).

Altheide and Carvey (2011) make a distinction in terminology between evidence, which is a legal construct and is presented in a court room or other official proceedings, and an artefact, which is a piece of data that pertains to an alleged action or event and is of interest to a digital forensic practitioner. Other researchers, (Carrier, 2005; Jarrett et al., 2009; Cohen, 2008) choose to use the term evidence inclusively, and this is the term used throughout this proposal.

The purpose of collecting digital evidence and physical evidence is the same; it allows legal practitioners and investigators to preserve evidence and then analyse it to determine the facts of an alleged crime or other unauthorised behaviour (Saferstein, 2011). Evidence can be categorised depending on the kind of argument it supports; inculpatory evidence is that which supports existing hypotheses, such as the guilt of a suspect, and exculpatory is that which contradicts the existing hypotheses or offers an alternative argument (Carrier, 2003).

Notable differences exist between physical evidence and digital evidence and in particular the volatile nature of digital evidence by virtue of the ease with which it can be altered (Caloyannides, 2004; Mercuri, 2005). This susceptibility to alteration can make it difficult for courts to determine the admissibility of some of digital evidence exhibits tendered during legal cases (Akester, 2004). For this reason it is important that when acquiring digital evidence the original data remains unaltered, a requirement that is met through the application of sound acquisition processes (Casey, 2011).

Digital evidence is also rich in metadata, which is information about the data that is stored on a device, such as the of a file size in bytes, the time of the most recent modification to the file, the last time a file was edited, or the time the file was initially created (Carrier, 2005, p.130). This metadata can assist the digital forensic practitioner in establishing timelines of events and identifying forensically interesting artefacts within the data (Buchholz & Spafford, 2004, pp.4-5).

The analysis of digital evidence routinely involves working with large data sets that often use new and emerging technologies that require technical expertise and understanding that the typical legal practitioner does not possess (Mercuri, 2005). This technical complexity can lead to fundamental misunderstandings of the evidence presented in legal cases, resulting in flawed reasoning and justifications when considering a verdict (Koehler & Thompson, 2006).

2.2 Digital forensic processes

The motive and desired outcome of a digital investigation may vary depending on the environment, individuals involved, and the nature of the behavior. However, the general forensic principles and processes do not, as it is vital that evidence must be acquired, preserved, and analysed before conclusions can be drawn from it (Altheide & Carvey, 2011).

The acquisition and preservation processes involved in digital forensic examinations have different objectives but are closely linked (Casey, 2011). Preservation must be considered by a practitioner before the acquisition process begins, as the acquisition process itself can alter the source device (Sommer, 1998). Preservation techniques, therefore, must be employed during the acquisition of the data to prevent contamination and preserve the data in a condition identical to that it was seized in (Hosmer, 2002).

Once a practitioner has collected and suitably preserved the digital data in a manner that is assured of later proving its validity, the image may be analysed to complete the forensic examination (Carrier & Spafford, 2003). There are three main kinds of digital forensic analysis:

- Media analysis is the identification, extraction, and interpretation of digital evidence on storage devices like hard drives and USB flash media (Digital Forensics Research Workshop, 2001),
- Network analysis focusses on traffic and communications on networks between computers (Corey, Peterman, Shearin, Greenberg, & Van Bokkelen, 2002), and
- Code analysis is the breaking down of suspect applications and programs to determine the composition and underlying mechanics that may be of interest to a practitioner (Aquilina, Casey, & Malin, 2008).

All investigations should start with a hypothesis to explain events and artefacts located in the data and the practitioner should then search for inculpatory evidence and exculpatory evidence (Carrier

& Spafford, 2003). These hypotheses may need revising as the investigation progresses and new evidence is detected, and should continue to be revised until a logical explanation of events is found (Carrier, 2005, pp.12-14).

Carrier and Spafford (2003) explored digital forensic processes and developed a model for investigating digital crime scenes called the Integrated Digital Investigation Process (IDIP). One of the primary tenets of this process is to ensure that any computer encountered in a physical crime scene is treated as a separate crime scene in terms of both the physical hardware and the digital evidence it may contain.

According to Carrier and Spafford (2003), the IDIP model prescribes 17 phases organised into five groups. These groups are readiness, deployment, physical investigation, digital investigation, and review, and are described as follows:

- The Readiness phases are ongoing, and are designed to maintain a forensic capability at all times.
- The Deployment phases involve the detection of suspect events and obtaining authorisation to pursue an investigation into those events.
- The Physical Investigation is designed to secure, locate, document, collect, reconstruct and present those physical artefacts within the digital crime scene.
- The Digital Investigation undertakes the same phases but in relation to the data on devices collected during the Physical Investigation.
- The Review phases are retrospective and allow the practitioner to reflect on the process and results of the investigation to identify areas of improvement.

Carrier and Spafford (2003) note that a digital crime scene does not only encompass the digital evidence, but also includes all the physical evidence associated with the computer. This may include removable devices, network cables, storage devices, printers, and even post-it notes attached to a device or other documentation pertaining to the computer (Ashcroft, 2001, pp.9-22). It is up to the

practitioner to use both the physical and digital evidence to connect the events that occurred on a suspect computer back to one or more users (Carrier & Spafford, 2003).

An alternative, and considerably newer, model is the Advanced Data Acquisition Model (ADAM), as designed and proposed by Adams (2012, pp. 152-155). According to Adams (2012), the ADAM's overriding principles focus on maintaining the integrity of the original data, keeping complete, detailed records of the state of the data and physical devices, and protecting the rights of all parties involved in the investigation. The model is divided into three stages; initial planning, creating an onsite plan, and acquisition of digital data (Adams, 2012). According to Adams (2012), these stages consist of:

- The initial planning stage, which involves identifying the requirements of a task, confirmation of authorisation to act, considering the case specific constraints, such as physical access and time limitations, and including this information in an outline plan.
- The creation of an onsite plan; this is done by assessing the specific characteristics of the operating environment, identifying and securing the required access to the data to be acquired, and making the necessary changes to the outline plan.
- The consideration of technical issues when acquiring the data, and the mandating of comprehensive note taking, creation of multiple copies of the acquired data, and ensuring that the integrity of the data can be verified.

Unlike the IDIP, the ADAM does not cover the analysis of the acquired data and presentation of any discovered evidence.

Of these processes, the acquisition and preservation of digital evidence is the focus of the next section, which examines these processes in more detail, covering the purpose, technical processes, potential complications, and mitigation for some of those complications.

2.3 Preservation and recovery of digital evidence

The preservation of digital evidence is intended to ensure that both the original device and any images acquired from that device remain in a pristine condition, meaning they are representative of the data stored on the device at the time of seizure (Casey, 2011). Effective preservation, as documented by acquisition logs and hashes¹, allows a practitioner to attribute events and items to the users of the device, and precludes any suggestion that they were generated by the practitioner themselves (Carrier, 2005). A key factor in preservation is the promptness with which a seized device has its data forensically acquired, with speed being crucial to preventing claims of deliberate or accidental contamination (Altheide & Carvey, 2011). The next section describes the acquisition and recovery processes in detail, and explores some potential sources of contamination and their counter-measures.

2.3.1 The role and importance of crime scene preservation

Evidence acquisition and preservation is required to ensure the continuing integrity (the objective truthfulness) of any evidence recovered from a device and allows it to be admitted during legal proceedings (Altheide & Carvey, 2011). The integrity of the original evidence directly affects its value and any forensic images relating to it in court; if a party cannot prove that evidence is unaltered then it can be implied by opposing legal counsel that it may have changed and is no longer an accurate representation of the actions of the accused (Noblett et al., 2000; Duerr et al., 2004). Therefore, sound procedure is required to prove that the investigative process has not altered the original evidence (Noblett et al., 2000), and acquisition is an important step in that procedure.

Assessing the suitability of evidence for presentation in a court of law is at the core of forensic science (Digital Forensics Research Workshop, 2001). The US Federal Rules of Evidence dictate the

¹ A hash is a fixed length representation of an arbitrary input. Identical inputs to a hashing function will produce identical hashes, meaning a hashing function can be used to prove two files are identical (Roussev, Chen, Bourg, & Richard, 2006). Hashing is further discussed in section 2.3.3.1.

requirements that must be met regarding the collection, preservation, and presentation of evidence in United States Federal Courts, including the Supreme Court of the United States (SCOTUS) (Federal Rules of Evidence, 2010). Given the scientific and methodical nature by which a digital forensic investigation is performed, and the non-traditional nature of digital evidence, rules 702, 901a, 1001(4), and 1002 of the US Federal Rules of Evidence are particularly pertinent to the digital forensic practitioner.

The equivalent Australian legislation is the Commonwealth Evidence Act 1995, with each state and territory having its own Evidence Act (Mason, 2007). Moles (2007) notes that Australian courts have acknowledged *Daubert vs Merrell Dow Pharmaceuticals Inc.* (1993), the current precedent in US Federal Courts (Gottesman, 1994), in assessing evidence presented by expert witnesses, such as in *Osland v The Queen* (1998). It should be noted that Australian evidence requirements are far less prescriptive and rely heavily on the magistrate or judge to analyse the circumstances surrounding the acquisition, analysis, and presentation of digital evidence (Moles, 2007).

Looking at the US experience, to meet the requirements of the US Federal Rules of evidence, the integrity of the data cannot be in question; the data structures must have remained the same throughout the investigative process (Digital Forensics Research Workshop, 2001). This requirement is primarily driven by rules 1002, 'Requirement of the Original', and 1001(4), 'Definitions that Apply to This Article' (Jarrett et al., 2009). Rule 1002 requires that to prove the content of an item of evidence, the original must be provided (Federal rules of evidence, 2010); however, due to the nature of digital evidence, presenting the original storage device would be ineffective (Thompson & Berwick, 1998).

Rule 1001(4) allows for copies where appropriate, such as a photograph developed from a negative, so long as it is, "*. . . produced by methods possessing accuracy which virtually eliminates the possibility of error*" (Federal Rules of Evidence, 2010). These copies are considered best evidence; the courts do not consider there is a more appropriate, feasible method of preserving and

presenting digital evidence, especially multimedia such as digital photos, video or audio (Thompson & Berwick, 1998)

2.3.1.1 The acquisition process

Forensic acquisition is a process intended to capture an accurate (objectively truthful) and independently verifiable representation of data stored on physical evidence-and which can be validated in that it may be relied upon as evidence in proceedings (Altheide & Carvey, 2011). A sound acquisition process creates an exact duplicate of a suspect data storage device, a 'bit for bit copy', or a specialised forensic container holding the equivalent thereof (Lyle, 2003). This process must be well documented to prove the provenance and integrity of the captured image, repeatable with the same results to allow for authentication as digital evidence, and capture a forensic image that is a complete and unaltered representation of the data stored on a device (Noble et al., 2000; Altheide & Carvey, 2011). These processes are designed to protect the integrity of digital evidence. Ideally acquiring and verifying the integrity of a forensic image should never modify the source media (Duerr et al., 2004).

The initial captured forensic image, sometimes called the 'master image' should be copied to create one or more 'working' backups and to provide a copy for the opposing counsel (Pollitt, 2001). This means that the practitioner can later analyse a perfect representation of the original evidence without running the risk of corrupting the original image (Digital Forensics Research Workshop, 2001). Using an imperfect image for analysis would diminish the admissibility as well as placing at risk the evidentiary worth of evidence recovered from the device (Noble et al., 2000; Carrier, 2005).

Carrier (2005) explains that data can be acquired at different levels, such as the whole disk, a single partition, or a single file, depending on what the practitioner is attempting to preserve. Accordingly, a practitioner should make an informed decision prior to creating a forensic image. Usually this will

be the lowest level possible, the whole disk, as it ensures that all possible data has been collected and can be analysed (Carrier, 2005).

Acquiring a forensic image at a higher level will inherently exclude some data, so the practitioner must be confident that doing so is well justified and is cognisant of exactly what is being excluded; this is called selective imaging (Turner, 2006). For example, Carrier (2005) suggests that when acquiring data from an intrusion detection system (IDS), it may be more appropriate to capture specific log files only, if the IDS is not thought to have been compromised. Other considerations in selective imaging may be the capacity of the practitioner to acquire large volumes of data or legal requirements in regards to legally privileged information, such as medical or legal documents (Turner, 2006).

The acquisition process itself differs in detail depending on the forensic tools used, however, a basic explanation can be extracted from the US National Institute of Standards and Technology (NIST) testing guidelines (Lyle, 2003). The first step of the acquisition process is to record the details of the source, the environment it was found in, and whether it was removed from the original host machine (Jarrett et al., 2009). Typically a practitioner would now connect a software or hardware write-blocker before creating a hash of the disk (Carrier, 2005), though NIST neglects to do this so that the acquisition software can be tested to determine if it writes to the source drive (Lyle, 2003). The forensic image acquired needs to be stored, so the destination media should be prepared: formatted and partitioned for a forensic image, or left blank for a bitwise clone (a technique that copies data from one disk to another, such that every bit ends up at the same address on the copy, as opposed to an image which groups all the data into a single file; Carrier, 2005). Carrier (2005) suggests writing zeros to the drive beforehand if using the latter case, to make it clear where the captured data ends and to prevent contamination from other data previously on the disk.

Once this preparation is completed, the capture of the target data can begin using the forensic tool chosen. The capture process progresses methodically, reading a chunk of data and writing that

chunk to the destination drive, then repeating the process until the entire source drive has been copied (Carrier, 2005). NIST (Lyle, 2003) recommends finishing the acquisition process by computing the hashes of the image and comparing them to those calculated at the beginning of the acquisition process to verify whether they are identical.

The alternative to acquiring an image of a device for investigation in an environment in which the original computer system is powered off (sometimes referred to as dead analysis) is live analysis which uses the hardware and operating system of the suspect computer to perform the digital forensic investigation, rather than an environment completely controlled by the practitioner (Carrier, 2006).

Carrier (2005) identifies numerous negative aspects to the live analysis, notably:

- the actions of the practitioner can irreversibly alter the data, making it hard to differentiate between their actions and those of a potential suspect
- the OS and hardware can actively work to hide data from the practitioner, such as HPAs and DCOs², and
- as the suspect controls the environment, it can make the practitioner more susceptible to triggering evidence destroying programs, including digital booby traps.

Carrier (2005) concludes that while a live analysis should be avoided where possible, there are times where a live analysis is the only option, such as when dealing with critical systems that cannot be taken offline for acquisition, or when knowingly dealing with full disk encryption.

2.3.1.2 Chain of custody

The chain of custody is a concept and process designed to ensure the integrity of evidence, including digital evidence (Cosic & Baca, 2010). The process requires those holding custody over evidence to

² Host Protected Areas (HPAs) and Device Configuration Overlays (DCOs) are areas of a storage that cannot be read by the file system. They are discussed in more depth in the section 2.3.2.3.

be able to provide a clear timeline highlighting the location, state, and significant events as they pertain to a piece of evidence (Cosic & Baca, 2010). The chain of custody can be a point of challenge for opposing counsel and for a practitioner trying to introduce digital evidence in a court; defending a challenge can hinge on the procedures used to document and protect the forensic image (Berg, 2000). However, the possibility of alteration is not enough to make evidence inadmissible; there must be credible evidence that alteration has taken place to render it inadmissible, though the identification of inadequate evidence protection processes can reduce the weight of that evidence (Jarrett et al., 2009).

Digital evidence needs protecting in terms of the data, which is the focus of this review, and the physical media on which it is stored (Jarrett et al., 2009). A two stage process involving a documented chain of custody for the physical evidence and a comprehensive set of hashes for the data held by the device, is mandatory and ensures the continuing integrity of the digital evidence (Jarrett et al., 2009).

Cosic and Baca (2010) suggest that documenting the chain of custody should involve addressing the following questions and recording the information for future reference:

- What is the evidence?
- How did investigators get the evidence?
- When was it collected and when was it used?
- Who handled it?
- Why that person handled it?
- Where it travelled, where was it stored?

The overall goal of any procedure that tracks the information required to answer these questions is to prevent unauthorised persons from having access to the evidence between the time of capture and presentation in court (Cosic & Baca, 2010). It is essential to record the chain of custody as near to the time and place of collection as possible, and this should include the identification information

of the practitioner generating the data (Duerr et al., 2004). The collection of this information should be complete and detailed, making clear who has accessed the evidence, when, what was done to it or what it was used for, and when it was returned, and ideally should follow a documented process (Turner, 2005).

2.3.2 Sources of contamination and preservation issues

Locard's Exchange Principle explains how the perpetrator of a crime will bring something into any crime scene and will leave with something from it (Saferstein, 2001). Carrier and Spafford (2003) explain how this principle applies to a digital crime scene: any action an individual takes when electronically interacting with a digital crime scene will alter the system if specialised equipment and procedures are not used. This includes the actions of the practitioner, and is a strong reason why live analysis can diminish the admissibility and reliability of the evidence; the practitioner can inadvertently alter the digital crime scene, compromising its integrity as evidence, as all artefacts can no longer be guaranteed to be a historical account of events taken place on the system (Carrier, 2006). The acquisition process is designed to avoid the transference of evidence from the practitioner to the storage device (Sommer, 1998).

Contamination refers to the addition or alteration of data, by the practitioner or another person, as well as the loss or uncontrolled, non-capture of data (Gupta, Hoeschele, & Rogers, 2006). Acquisition equipment needs to be capable of capturing all the information present on a disk (Lyle, 2003), which includes the HPA and DCO³ areas (Gupta et al., 2006). While the potential loss of evidence present in any uncollected data is a setback for any investigation, the more important implication is that if the omission of data from a forensic image is discovered, it can damage the credibility of all findings and testimony produced by the practitioner based on that image (Carrier, 2005; Cohen, 2008).

³ Host Protected Areas (HPAs) and Device Configuration Overlays (DCOs) are areas of a storage that cannot be read by the file system. They are discussed in more depth in section 2.3.2.3.

2.3.2.1 BIOS addressing limitation

Different acquisition tools access the hardware in different ways, either through the Basic Input/Output System (BIOS), or through direct connection to the media (Carrier, 2005). Carrier (2005) notes that the BIOS can potentially report incorrect information, either through technical limitations, firmware errors, or possibly deliberate obfuscation caused by custom firmware. For example, a BIOS using a 32 bit addressing scheme can only detect 2.2 terabytes of a 3 terabyte hard drive, and in turn will only report 2.2 terabytes as the size to acquisition software using the BIOS (Microsoft, 2013).

Computer viruses running on the practitioner's computer, either extant or contracted through interaction with a suspect device, also have the potential to alter or destroy data randomly or in a targeted attack (Berg, 2000).

2.3.2.2 Bad sectors

Bad sectors and other input/output (IO) errors encountered during the acquisition process should be logged in detail (Lyle, 2003). Bad sectors are a part of a storage device that has become corrupted, either through hardware malfunction or a software error in writing data (Carrier, 2005). When a bad sector is encountered during the acquisition process most software will record the entire sector as being filled with zeros, called benign fill by Lyle and Wozar (2007), rather than the corrupt data, in a case of deliberate contamination (Altheide & Carvey, 2011). While inserting benign fill results in an image that is not a perfect copy of the source media, writing a fixed number of zeros ensures the correct number of bytes is written to the image, keeping the overall size and relative structure of the image consistent with the source media (Carrier, 2005). Bad sectors have been shown, however, to cause the popular Unix image creation tool dd to omit up to six sectors following a bad sector, resulting in an image that has not captured all available data (Lyle & Wozar, 2007).

2.3.2.3 Hardware level data obfuscation

Host Protected Areas (HPAs) and Device Configuration Overlays (DCOs) are areas of a hard drive or solid-state drive that are protected from reading by the operating system (Guo, Slay, & Beckett, 2009). These areas are used by vendors to store data, such as drivers and recovery information, without it being at risk of modification or erasure; however, these areas can equally be used by an individual to store other data that they do not want found by practitioners or other users of the computer (Gupta et al., 2006). During the acquisition process the practitioner can use tools to detect and capture these areas on a hard disk or solid state drive and subject them to analysis (Altheide & Carvey, 2011).

2.3.3 The importance of forensic image validation and preventing contamination

The case the literature makes is that imaging is required to be an errorless process by which a copy can be obtained of original evidence. When dealing with digital evidence it is not immediately obvious that the copy is a true and accurate reproduction, and therefore relies on the expert testimony of the practitioner to provide validation (also called authentication) (Jarrett et al., 2009). Validation is a process by which a practitioner establishes that inferences drawn from the digital evidence can be reproduced and verified and are a suitable basis from which to form legal arguments (Casey, 2011).

Where arguments have been presented to the court but cannot be verified or have failed verification, the case often collapses for the party leading that argument, especially if the failure is a result of professional incompetency (Cohen, 2008). Similarly, a case is also in jeopardy should the opposing party be able to show that the security of the environment in which digital evidence was stored was lax and that contamination of the evidence resulted from this shortcoming (Cohen, 2008). In cases where the evidence has been validated and accepted by the court but questions remain about the reliability of the evidence, the decision makers in that case may give the evidence

less weight (Casey, 2011); they will not consider the evidence and associated arguments to be a definitive source of truth.

Rule 901 of the US Federal Rules of Evidence, Authenticating or Identifying Evidence, states that the onus is on the proponent of the evidence in question to “. . . *provide evidence sufficient to support a finding that the item is what the proponent claims it is*” (Federal Rules of Evidence, 2010), a point reinforced by Duerr et al., (2004) and Berg (2000). Jarrett et al. (2009) also state this, but make the distinction that only the process and forensic image must be sound to achieve admissibility, while the accuracy of the information originally on the device is a matter for argument at trial.

Jarrett et al. (2009) elaborate that evidence can be authenticated by a person with demonstrated knowledge, “. . . *that a matter is what it is claimed to be . . .*”; meaning that a person present during the seizure and acquisition of digital evidence is qualified to assert that the correct process has been carried it. Jarrett et al. (2009) state that the person need not be a forensic expert and give the example of a FBI Agent not trained in forensic acquisition being qualified to authenticate the evidence.

To assist practitioners in producing evidence that meets validation standards, the National Institute of Standards and Technology (United States) runs the Computer Forensics Tool Testing (CFTT) Program. This programme is designed to provide assurance to practitioners and legal practitioners as to the accuracy and validity of the data generated by digital forensic tools, using the scientific method (Lyle & Wozar, 2007). The results of NIST testing allow evidence to be presented in court without the requirement to prove the process and tools used to create it are accurate, thus reducing the likelihood the validity of the evidence will be challenged by opposing counsel or ruled inadmissible by the person in charge of proceedings (Guo et al., 2009). The tests start from a set of base requirements that every tool must achieve (Lyle, 2003):

- it must not alter the original disk,
- it must make a perfect duplicate,

- both of these requirements can be independently verified, and
- it logs any disk errors it encounters.

In addition to the mandatory criteria, there exists a range of optional criterion to allow for the testing of specialised features of different tools (Lyle, 2003).

More recently the UK Forensic Science Regulator released a draft Digital Forensics Method Validation guide focussing on the validation of the processes used to procure digital evidence (2014).

The guidance provided instructs practitioners to ensure that all processes they use are shown to produce reliable evidence before being used in real cases and that adequate documentation is created and retained detailing the steps followed to validate their processes. These processes include the digital forensic tools used, but they are described as only part of the validation regime, a deviation from NIST standards which focus almost exclusively on the technical aspects of process validation. The Digital Forensics Method Validation guide requires the use of multiple software tools using fundamentally different code bases to ensure that the tools used provide a consistent output regardless of the technology used to implement the functions (2014). However, as with NIST guidelines there are no specific criterion addressing the integrity of the case files and chain of custody, despite them being an important part of digital forensics processes.

2.3.3.1 The role and importance of hashing and digitally signing the forensic image

Once created, a forensic image needs to be verified as being an accurate copy so as to be authenticated and considered admissible; this is achieved through a mathematical process called hashing (Jarrett et al., 2009). Hashing involves taking an input of any length, performing complex mathematical operations on it, and outputting a fixed length string that is intended to be unique (Noblett et al., 2000). The modification of a single bit of the input will result in an altered output, and the process will enable detection of any changes to the target data by comparison of the two inputs (Altheide & Carvey, 2011). Two identical inputs, such as two image files, will produce identical hashes, supplying a repeatable way to prove that two files are identical, as it is computationally

infeasible for non-identical inputs to produce the same hash (Roussev, Chen, Bourg, & Richard, 2006).

Using hashing to verify a forensic image involves taking multiple hashes at different points during the acquisition process (Altheide & Carvey, 2011). The first step is to take a hash of the suspect device, to create a benchmark against which it can be proven that the original has not been altered by the process (Noble et al., 2000). In devices containing HPA or DCO areas it is advisable to take two preliminary hashes; before and after enabling these protections (Carrier, 2005). After this hashing and verification process the forensic image is acquired and both the original media and the image are again hashed; this new device hash is used to prove that the acquisition process has not altered the original in any way and the image hash is used to prove that the forensic image is an identical representation of the data present on the device (Roussev et al., 2006).

Some forensic acquisition tools will hash larger chunks of data as the process progresses (Altheide & Carvey, 2011). The benefit in this behaviour is that it can be used to prove the integrity of a certain part of a forensic image where the rest has been contaminated or otherwise invalidated (Roussev et al., 2006). If the acquisition process is repeated it can be shown that individual chunks of the same size and cluster range remain identical, even if others have changed (Roussev et al., 2006).

Some older hashing functions, such as MD5, have been proven to have collisions, that is non-identical inputs that generate identical hashes thus putting the reliability of the hash in doubt (Wang & Yu, 2006). While these are rare, and have only been demonstrated in controlled, deliberate circumstances, the vulnerability can be mitigated by providing hashes using different hashing functions, such as MD5 and SHA1, as the likelihood of both hashes being collisions is computationally infeasible (Altheide & Carvey, 2011).

Carrier (2005) asserts that embedded hashes in specialised digital evidence containers⁴ do not increase the security or integrity of the image file, as anyone seeking to deliberately modify the forensic image could simply recalculate the hashes with the altered input. An encrypted file could also be altered by anyone with access to the decryption key (Oppliger & Rytz, 2003). Carrier (2005) suggests that one way to protect embedded hashes would be to use a cryptographic signature that included a timestamp to make it obvious that the forensic image has been altered after the original process, though such a process would require a trusted time source, which is not something a practitioner is likely to have access to during an offsite acquisition of data.

Hashing is the current standard for proving the integrity of forensic images and is considered by the US Department of Justice as a means of proving the identical nature of two files or images (Jarrett et al., 2009), however, the hashes must be protected and are subject to simple alteration that may be difficult to detect (Hosmer, 2002). An alternative is to use digital signatures, which bind the known identity of the practitioner, as verified by a third party Certificate Authority (CA), an independently verified timestamp, and the data together (Hosmer, 2002). This approach still uses a hash, except that it is embedded in the signature; the verifier can still access the hash, but can now be assured of who calculated it and when (Oppliger & Rytz, 2003). The signature cannot be altered, though a new one can be generated; this new signature will have a new timestamp (Oppliger & Rytz, 2003).

According to Hosmer (2002), digitally signing hashes or logs is an improvement on simple hashing, but has a number deficiencies, notably:

- the private keys used to create the signatures could be compromised, allowing others to generate signatures in the practitioner's name;
- the certificate proving the practitioner's identity could expire, making all previously signed signatures invalid;

⁴ Digital Evidence Containers (DECs) are constructs, which not only include the files from the source device, but also logs, hashes, and other information about the acquisition process. DECs are further discussed in section 2.3.3.3.

- the public keys or certificate could be lost, meaning the existing signatures could not be read; and
- the signature still does not bind to the data, there is no guarantee that the hash represents what the practitioner claims it does.

Duerr et al. (2004) investigate the possibility of using digital certificates to help prove the chain of custody for digital evidence by signing the results of major events in the chain of custody. Duerr et al. (2004) propose attaching a digital certificate to any information that is added or altered, such as the various hashes generated throughout an investigation, creating a sequence of signatures describing the life of the evidence. This would allow a practitioner to say with certainty that any hash presented was calculated at a specific time and by them (Duerr et al., 2004).

2.3.3.2 Tools available to prevent contamination

Preventing contamination of original evidence is typically done using hardware or software write blockers (Lyle & Black, 2005). These write blocking processes have the same end goal of preventing the practitioner from writing to the source media during the acquisition process, but approach it in different ways (Lyle & Black, 2005). Despite their widespread use and stated purpose, Carrier (2005) notes that neither are immune to sophisticated booby traps designed to destroy, alter, or obfuscate data on a drive being acquired.

Hardware write blockers are devices that act as a gateway between the practitioner and the storage device being acquired (NIST, 2004). The source device is connected directly to the write blocker while the practitioner operates the write blockers, either from a console or through buttons on the write blocker (Carrier, 2005). The write blocker allows the practitioner to acquire systematically a forensic image and ensure that the storage device being acquired is not altered in any way (Lyle & Black, 2005). The write blocker achieves this by screening incoming instructions directed at the device and removes or disables any instructions it identifies as potentially causing changes

(Hardware Write Blocker Device (HWB) Specification Version 2.0, 2004), thereby disallowing any write requests made of the device thereby preventing contamination.

Software write blockers have the same goals as hardware write blockers but achieve this in a fundamentally different way (Lyle & Black, 2005). Rather than directly connecting to the device a software write blocker is loaded on a computer the device is connected to; often the computer storage device has been used with (Carrier, 2005). The software is loaded through some external device, such as USB flash memory or a DVD containing a bootable forensic imaging environment, directly into memory. As the software loads it alters the interrupt table of the computer that causes the BIOS to execute custom code when it receives an instruction targeted at the storage device, rather than native BIOS code (NIST, 2003). This custom code monitors incoming instructions and strips out or alters any instructions that would potentially alter the data or configuration of the device, to produce an effect similar to that of a hardware write blocker (Lyle & Black, 2005).

Device Configuration Overlays and Host Protected Areas can be troublesome for write blockers, as they require alteration of the device to be acquired to make them accessible, the exact thing they are designed to prevent (Gupta et al., 2006). Most write blockers have been modified to facilitate these instructions, or allow them to be toggled on or off, though older write blockers may not have this functionality and would provide an incomplete forensic image (Carrier, 2005).

2.3.3.3 Forensic containers

When a forensic image is acquired it can be stored in a number of formats including raw format, a bit for bit copy of the acquired device contained within a single file, or in several common forensic evidence containers (Altheide & Carvey, 2011; Carrier, 2005). These digital evidence containers (DECs), combine the data collected during acquisition with extra supporting information, such as metadata about the image and acquisition logs, into a single structure (Richard et al., 2007). The raw format, a file which simply contains the data from the acquired device and nothing more, is commonly used and capable of being opened and viewed by most forensic tools (CDESWFG, 2006),

DECs offer additional benefits to the practitioner, but may be proprietary or closed source (Carrier, 2005). DECs, such as EnCase Evidence File (E01) and Advanced Forensics Format (AFF4), are files which contain the unaltered data from the acquired device and offer additional features that can make storing, protecting, transporting, and analysing the data an easier task (CDESWFG, 2006). Typical functionality includes automatic checksums and hashing, compression, encryption, and splitting (CDESWFG, 2006). E01, one of the most common formats, and AFF4, support all of these functions, with AFF4 also supporting arbitrary metadata fields for use as the practitioner sees fit to choose (Altheide & Carvey, 2011).

Automated checksums and hashing features of these forensic acquisition tools assist examination as they calculate the hashes for the source device and the forensic image automatically, which can be stored with or within the image file, and are intended to prove whether the original evidence has been altered or remains intact (CDESWFG, 2006). Most tools provide compression of the image to reduce storage size and make them more efficient to store and transfer, which is useful when acquiring multiple high capacity devices, but may result in longer acquisition times (CDESWFG, 2006). Encryption features allow a practitioner to protect forensic images from improper access by making them unreadable without the encryption key, thereby preventing alteration of the image and affording added protection to important data within the image (Altheide & Carvey, 2011).

However, these digital evidence containers do not protect the contents from alteration (Carrier, 2005). Other researchers have identified this issue, and similar shortcomings, and discuss the current focus of research efforts and identify research areas requiring increased development.

2.4 Research Directions

A decade ago researchers noted that much of the contemporary research of the time was geared towards providing practitioners with additional and improved means to locate and preserve digital evidence (Lenstra & de Wegner, 2005; Schneier, 2004). Garfinkel (2010) considers that the current suite of digital forensics tools are focussed on evidence discovery, rather than as tools to assist in the

execution of a broader investigation. Garfinkel (2010) suggests the need for an enhanced preservation process geared towards improving the capabilities and efficiency of tools. Garfinkel (2010) opines that the current operational ability of practitioners will diminish over time as the complexity and breadth of devices holding digital evidence increases and calls for a reorientation of research focus away from purely technical challenges, to addressing greater issues within the science and processes of digital forensics. Garfinkel (2010) suggests that standardised processes for data acquisition be developed for a wide variety of electronic devices, such as ebook readers and mobile phones, and that tools be developed to assist in cases where a computer was used to commit a crime, but does not contain criminal material.

Mocas (2004) suggested that two worthwhile research areas are data integrity and authentication, topics which deal with processes and developments that are mandatory to ensure the continuing admissibility of digital evidence in a court of law.

Garfinkel, Farrell, Roussev and Dinolt (2009) assert that the results of digital forensic research experiments to date is not entirely reproducible by other researchers; while similar results can be obtained using the published methodology of the original authors, the use of different experimental data (i.e. the data on the mock suspect hard drives used in the experiments) means that the results cannot be reproduced exactly, weakening the scientific merit of any results and conclusions drawn from the experiment. To combat this a freely available digital forensics corpus (available at digitalcorpora.org) consisting of files and assorted forensic images, including images taken of 'real world' user devices has been created (Garfinkel et al., 2009). Researchers can select data from this repository to populate the devices in their experiments, which means that anyone wishing to reproduce the experiment can download and use exactly the same data in their reproductions as were used in the original.

2.5 Summary

Digital evidence is a complex and fragile form of evidence (Caloyannides, 2004) that requires specialist practitioners to acquire, preserve, and analyse it (Rogers et al., 2007). Where digital evidence is required to be used in court proceedings it must be validated, which means it needs to be proven to be an accurate representation of the data held on the original device (Altheide & Carvey, 2011). Standardised tools and processes are available to ensure digital evidence acquisition and preservation processes meet the validity requirements of the courts (Lenstra & de Wegner, 2005).

However, the artefacts used to establish validity to the courts may be vulnerable to alteration, either inadvertent or through tampering. Carrier (2005) asserts that embedded hashes in digital evidence containers does little to protect the validity of digital evidence and Hosmer (2002) claims that hashes are simple to alter. Oppliger and Rytz (2003) opine that encryption as a protective mechanism is vulnerable to a compromise of the encryption key.

Garfinkel (2010) and Mocas (2004) call for further investigation into the processes and scientific theory behind digital forensic practice, opining that for too long research has focussed on overcoming technical challenges. Lyle has produced a number of papers proposing methods for evaluating the performance of digital forensic acquisition tools, such as hardware write blockers (Lyle & Black, 2005) and write-blocking software (Lyle, 2003). Further analysis of commonly used digital forensic acquisition tools, especially in the context of the ability of practitioners to validate the evidence they produce, is proposed in subsequent chapters. The analysis is intended to better understand the shortcomings in digital forensics research identified by Garfinkel and Mocas. This will provide a strong base from which recommendations towards improvements to the validation process can be made.

3 Research objectives

The research described in this thesis aims to confirm and further understand possible defects in current digital evidence validation practices processes by testing the integrity of the common forensic container format Encase Evidence File, also known as .e01. The Encase Evidence File format was chosen as it is the most widely used digital evidence container. The time permitted to complete the research did not permit the examination of other, less commonly used imaging tools. The results are intended to provide recommendations towards the mitigation of weaknesses in the container format identified during the research experiments. Further development based on these findings may allow for the presentation of more robust digital evidence and provide the justice system with the resources it requires to protect the integrity of its decision-making processes.

3.1 Digital evidence container metadata

Of particular interest to legal adjudicators are the acquisition process logs and digital evidence container metadata generated by common forensic acquisition tools, such as FTK Imager and EnCase Forensic Imager. This metadata is used to show that a sound acquisition process has been adhered to and that the forensic image produced is a complete and authentic reproduction of the data stored on the physical device. As mentioned in section 2.2, the soundness of the acquisition process is supported by the calculation of hashes, which are verified to be identical with those calculated from the original device. Preliminary experimentation revealed that the metadata contained within an Encase Evidence File forensic container could be manipulated via the use of open source third party libraries, such as libewf. This raised concerns about the reliability of existing processes to determine the validity of the forensic images.

Alterations to this metadata could have applications limited only by the imagination of the perpetrator, who may be the practitioner themselves, or another person with access to those materials, such as a network intruder or a colleague, or opposing legal counsel. If these alterations

are possible, and more importantly, cannot be detected by the software tools that made them, then it is feasible that a forensic image could be altered and the accompanying metadata altered to make it appear as if the image was unchanged. This leads to a scenario where a forensic image could have data added or removed from it to influence the findings of the practitioner and ultimately the outcome of a legal case or other situation where the tampered evidence is used to reach a decision.

3.2 Research questions and hypotheses

The overall objective of the research is to ascertain the reliability of current digital evidence validation processes and the tools that support them. The heavy reliance upon digital evidence container reports, metadata, and hashes with little research into the effectiveness of their protection mechanisms makes digital evidence containers an important area of analysis. The research seeks to determine whether a misplaced reliance upon digital evidence containers might allow a malicious actor to alter digital evidence without detection and ultimately subvert the operations of the courts.

The research progressed in stages, depending on the outcome of previous tests. While the ultimate goal was to alter a digital evidence container such that it was impossible to detect the alterations, the research was inspired by the assumed limitations of the Encase Evidence File and the tools available. Accordingly, questions 1 and 2, and the accompanying hypotheses, were considered preliminary investigation, while question 3 and H_3 rely on H_1 and H_2 being unsupported. Question 4 was formulated once the limitations imposed by the technology, tools, and time available became more clear.

Each hypothesis was developed from its associated research question after identification of a viable technique for evaluating each premise, thereby providing the opportunity to develop test cases for the techniques, and facilitating scientific verification of the outcomes of the tests. This approach was selected because of the investigative nature of the research and the paucity of background research.

3.2.1 Alteration of digital evidence container metadata

Question 1:

Can the metadata embedded within digital evidence containers be meaningfully altered or modified?

H₁: The identifying metadata embedded within Encase Evidence File digital evidence containers cannot be meaningfully altered.

The first hypothesis is designed to establish whether the metadata used to identify a digital evidence container and its contents can be altered in a meaningful way. Being a file, it is a trivial task to alter a digital evidence container using a hex editor. However, due to the encoded nature of many proprietary file formats, it was likely that creating targeted, meaningful alterations would be more challenging, as the alterations would have to be made in specific locations within the file, or risk simply corrupting the original data with meaningless junk data. This question is intended to determine if and how alterations can be made to the original digital evidence container that may lead to an examiner making conclusions influenced by modified metadata fields, such as the date of initial evidence acquisition.

3.2.2 Alteration of forensic images within digital evidence containers

Question 2:

Can the content, the forensic image, contained within the digital evidence container be meaningfully altered?

H₂: The forensic image embedded within Encase Evidence File digital evidence containers cannot be meaningfully altered.

The second hypothesis is designed to establish whether the forensic image contained within a digital evidence container can be altered in a meaningful way. As with the digital evidence container, the proprietary nature of the digital evidence container was likely to impede the ability to simply alter the data within the forensic image. This question seeks to determine if and how alterations can be

made to the original forensic image contained within a digital evidence container that may lead to an examiner making conclusions influenced by modified digital images, such as the addition, removal, or modification of a digital photograph file.

3.2.3 Detection of alterations to digital evidence containers

Question 3:

Can meaningful changes to embedded metadata or forensic images be detected by digital forensic software tools?

H₃: Alterations to Encase Evidence File digital evidence containers cannot be detected by Encase Imager.

Question 3 relies on the previous questions establishing that meaningful changes can be made to the metadata or forensic image contained within a digital evidence container, or both. This hypothesis seeks to determine whether the tools provided by software vendors are able to determine whether the contents of a digital evidence container have been altered after the initial acquisition process. Examples of detection methods include automatic hashing and validation of the digital image and functions for validating an entire digital evidence container. This question seeks to identify the extent to which a malicious actor may be able to influence the forensic analysis and conclusions a digital forensic practitioner may reach.

3.2.4 Differentiation of forensic image containers

Question 4:

Is it possible to create two digital evidence containers that contain different forensic images but the same image metadata?

H₄: Given two Encase Evidence File digital evidence containers with identical metadata, except the forensic image hashes, but differing forensic images, it is possible to determine which container is an accurate representation of the originally acquired device.

This question developed as a way to demonstrate the implications of the findings of the previous three questions. It is used to determine whether a combination of the techniques used to assess the previous hypotheses can be used to create a potentially realistic threat model and what additional measures are required to defeat automated digital evidence container verification functions. The objective is to demonstrate that through the manipulation of digital evidence containers it may be possible to present altered evidence to a court or discredit evidence presented by other digital forensic practitioners.

4 Experiment design and results

The experiments designed for this research were used as a form of verification of processes devised to alter digital forensic containers and circumvent the error detection and validation tools provided by the vendor. Each hypothesis was tested on its ability to achieve certain specific objectives designed to demonstrate that the circumvention techniques were effective, permanent, and meaningful. Given that the means to circumvent a particular feature were unknown when commencing this research, the hypotheses were developed only once a viable candidate process was established.

Preliminary attempts at achieving the goals of the research questions involved directly opening Encase Evidence File digital evidence containers in a hex editor and attempting to reverse engineer the fields within the file. It quickly became apparent that while this may be possible, it was not a viable strategy given the time constraints and expertise of the investigators. An alternative approach was identified that used the open source library libewf as the basis for file manipulation. libewf is used to create, manipulate, and examine Encase Evidence File files in a Linux environment. The open source nature of the libraries allows for simple inspection and modification of the code base to allow non-standard operations to be executed against the Encase Evidence File digital evidence container. It is through libewf and modification of it that these hypotheses have been tested.

When testing for the validation of Encase Evidence File digital evidence containers Guidance Software's Encase Forensic Imager was used. This software was chosen as it requires no paid license and is produced by the same vendor that created the Encase Evidence File. Given the vendor's intimate knowledge of the source code and specifications of the Encase Evidence File, it was assumed that Encase Forensic Imager was likely to provide the most effective error checking and validation functions available.

4.1 Equipment

This section includes the major software and hardware used during the project.

Software:

- Kali Linux 1.0.9
- Microsoft Windows 7
- Microsoft Windows 8.1
- Guidance Software Encase Forensic Imager 7.09
- VMware Workstation 10
- WinHex 17.8
- Technology Pathways ProDiscover Basic 8.2.0.5
- libewf-20140608
 - experimental distribution ⁵
- nps-2013-canon1
 - A forensic image taken of a digital camera memory card
 - Acquired from Digital Corpora⁶

4.2 Testing environment

To facilitate the investigation of the research questions and testing of the hypotheses a standard environment was developed. To reduce cost and combat equipment and time constraints the entire experimental environment was virtualised. The host computer was running Windows 8.1 and used VMware Workstation 10 as the hypervisor. This host computer was used to store the results of each experiment. The experimental machines were a Windows 7 machine with WinHex, ProDiscover

⁵

<https://53efc0a7187d0baa489ee347026b8278fe4020f6.googledrive.com/host/0B3fBvztptiiSMTdoaVExWWNsRjg/>

⁶ www.digitalcorpora.org

Basic, and Encase Forensic Imager installed or present, and a Kali Linux machine with libewf version 20140608 installed. To ensure a sterile and consistent experiment environment each machine was restored to a standard snapshot of a clean install after each experiment. Neither experimental machine was connected to the internet.

The image used for testing was downloaded from Digital Corpora to ensure that researchers wishing to replicate this work have access to the exact same experimental data as was used in this research. The image used was named nps-2013-canon1, was distributed as an Encase Evidence File, and was originally acquired from a 128MB SD card used in a digital camera.

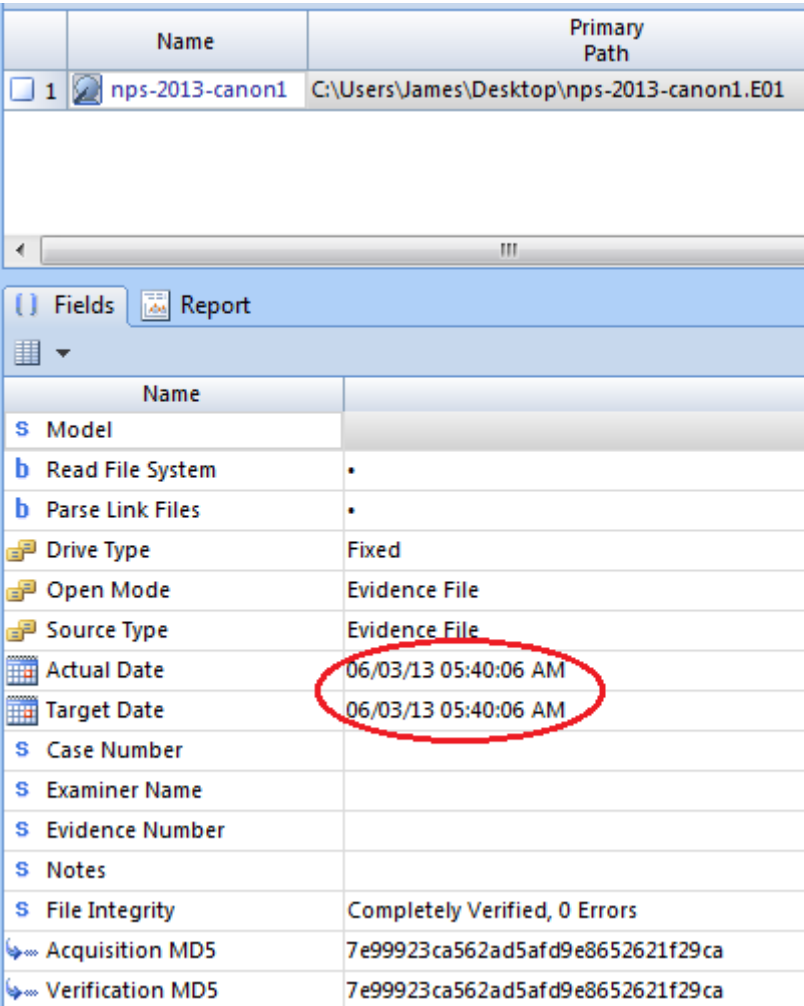
The digital evidence file itself holds the standard metadata any file in an NTFS (default Windows) or EXT4 (popular Linux) file system would: creation, last access and last modification dates. A range of tools are available that can trivially alter this metadata and accordingly altering this file metadata has been omitted from the experiments for simplicity.

4.3 Testing of H_1

H_1 : The identifying metadata embedded with Encase Evidence File digital evidence containers cannot be meaningfully altered.

This hypothesis (H_1) attempts to determine the ability of an individual to alter the metadata, without regards to detectability. Accordingly, the outcomes to be tested are permanency and meaningfulness. Permanency means that any alteration made to the digital evidence container must persist when the file is copied, redistributed, or opened on another machine. To test this outcome the file was altered on the Kali Linux machine and then copied to the host machine, then to the Windows 7 machine, where the modified metadata field was viewed with Encase Forensic Imager to ensure the changes were permanent. To test meaningfulness the initial acquisition date of the digital evidence container was modified, as an alteration to this information could influence the decision

making or conclusions of a court or digital forensic practitioner, and it is a field that never requires modification, as the acquisition date will never change.



Name	Primary Path
1 nps-2013-canon1	C:\Users\James\Desktop\nps-2013-canon1.E01

Name	
S Model	
b Read File System	•
b Parse Link Files	•
Drive Type	Fixed
Open Mode	Evidence File
Source Type	Evidence File
Actual Date	06/03/13 05:40:06 AM
Target Date	06/03/13 05:40:06 AM
S Case Number	
S Examiner Name	
S Evidence Number	
S Notes	
S File Integrity	Completely Verified, 0 Errors
Acquisition MD5	7e99923ca562ad5afd9e8652621f29ca
Verification MD5	7e99923ca562ad5afd9e8652621f29ca

Figure 1: The acquisition dates and hashes of the original E01 file.

The specific steps undertaken were:

1. Copied nps-2013-canon1.E01 to the Kali Linux desktop.
2. Executed the following command to set the acquisition date to midnight, January 1st, 2014.

```
Ewfxexport -m '2014 01 01 00 00 00' nps-2013-canon1.E01
```

3. Copied nps-2013-canon1.E01 to Windows 8.1 host machine.
4. Copied nps-2013-canon1.E01 to Windows 7 desktop.

5. Started Encase Forensic Imager and opened nps-2013-canon1.E01.
6. Recorded the contents of the File Acquired metadata field.

The original contents of the File Acquired field were "06/03/13 05:40:06 AM". After altering the field and transferring the file to an unrelated computer and opening the file in Encase Forensic Imager the contents of the field were "01/01/14 00:00:00 AM", suggesting the evidence was acquired at a much later date than its actual acquisition date. This supports the conclusion that metadata fields can be altered in a meaningful way and that any alterations are permanent. Therefore, H₁ is not supported. A more exhaustive project examining every individual metadata field would be advantageous towards strengthening this finding.

Name	
S Module Serial Number	
S Module Firmware Version	
S Write Blocked	
S Label	
S Serial Number	
S Model	
b Read File System	•
b Parse Link Files	•
Drive Type	Fixed
Open Mode	Evidence File
Source Type	Evidence File
Actual Date	01/01/14 12:00:00 AM
Target Date	01/01/14 12:00:00 AM
S Case Number	
S Examiner Name	
S Evidence Number	
S Notes	
S File Integrity	Completely Verified, 0 Errors
Acquisition MD5	7e99923ca562ad5afd9e8652621f29ca
Verification MD5	7e99923ca562ad5afd9e8652621f29ca

Figure 2: The acquisition dates and hashes of the original E01 file.

The prototype code used to modify the date of acquisition metadata field can be found in Appendix A.

4.4 Testing of H₂

H₂: The forensic image embedded within Encase Evidence File digital evidence containers cannot be meaningfully altered.

This (H₂) hypothesis attempts to determine the ability of an individual to alter the forensic image contained within the digital evidence container, without regards to detectability of the changes. The primary outcomes to be tested are permanency and meaningfulness. Permanency means that any alteration made to the forensic image within the digital evidence container must persist when the file is copied, redistributed, or opened on another machine. To test meaningfulness a specific image within the digital evidence container's embedded forensic image was modified, as an alteration to this information could influence the decision-making or conclusions of a court or digital forensic practitioner, and it is a field that never requires modification, as the acquisition date will never change.

In seeking a solution, it was found that libewf offered no solution to modifying the forensic image data directly. Part of the documentation and source code alluded to an experimental function that would allow such modifications, however, it was not part of the prebuilt binaries and attempting to build from source failed, as a much older version of libewf, 20090528 was required. However, it appears that this particular version of libewf is no longer available from any software repositories, meaning it was not possible to use this experimental function of libewf.

libewf does, however, offer a function to export an Encase Evidence File digital evidence container to a raw format file, stripped of all proprietary functions. This raw image can be altered simply with a hex editor and allows data to be added, removed, or altered. libewf can also acquire from a raw image, creating a brand new Encase Evidence File digital evidence container. This new digital

evidence container, however, has entirely different metadata, and therefore cannot be considered a modification of the original digital evidence container.

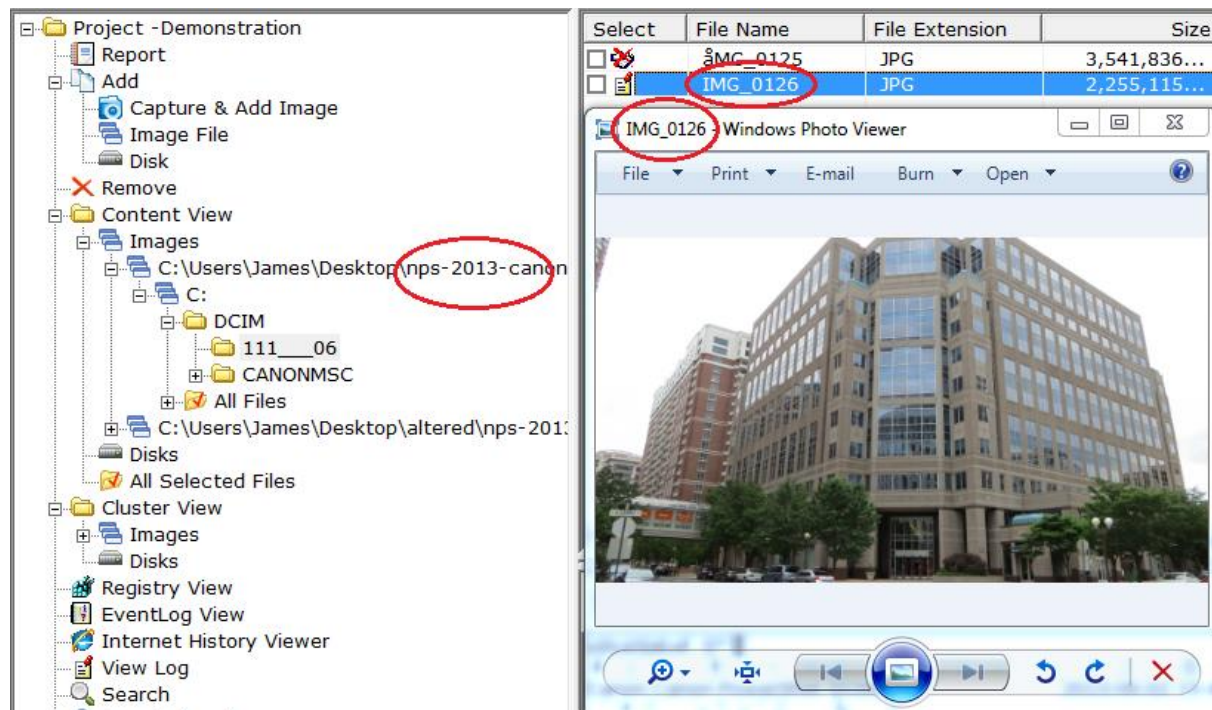


Figure 3: The original contents of the IMG_0126.JPG file.

To overcome this limitation the libewf acquisition function was modified such that an original Encase Evidence File digital evidence container can be provided as an argument when acquiring a raw image. This modified function saves the metadata from the original digital evidence container and overwrites the metadata fields of the newly created digital evidence container. The result is a digital evidence container that holds identical identification information to the original, and while not strictly a modification of the original, it is essentially a direct copy, with an altered digital image.

The specific steps undertaken were:

1. Copied nps-2013-canon1.E01 to the Kali Linux desktop.
2. Executed the following command to export the forensic image to raw format.

```
ewfexport nps-2013-canon1.E01
```

Named the output nps-2013-canon1.raw

All defaults were accepted in the following prompts.

3. Copied nps-2013-canon1.raw to Windows 8.1 host machine.
4. Copied nps-2013-canon1.raw to Windows 7 desktop.
5. Opened nps-2013-canon1.raw in WinHex and mounted as disk.
6. Browsed to \DCIM\111__06 and opened IMG_0126.JPG. Saved a copy of this file, a digital photograph of the US National Science Foundation building.
7. Noted the hex offset of the start of the file and securely wiped it.
8. Opened an image of a cartoon light bulb in WinHex and copied the hex values representing this image.
9. Pasted the hex representation of the light bulb illustration starting at the previously noted starting offset of the IMG_0126.JPG file. Saved the file, accepting warnings and prompts.
10. Copied nps-2013-canon1.raw to Windows 8.1 host machine.
11. Restored Kali Linux machine to original state and compiled and installed modified libewf.
12. Copied nps-2013-canon1.raw to Kali Linux desktop.
13. Copied nps-2013-canon1.E01 to Kali Linux desktop.
14. Executed the following command to acquire the modified raw image as an E01 with the original headers.

```
ewfacquire -9 nps-2013-canon1.E01 nps-2013-canon1.raw
```

Named output as ./altered/nps-2013-canon1.E01

All defaults were accepted in the following prompts.

15. Copied the newly created nps-2013-canon1.E01 to Windows 8.1 host machine.
16. Restored Windows 7 machine to original snapshot.
17. Copied the newly created nps-2013-canon1.E01 to Windows 7 desktop.

18. Opened nps-2013-canon1.E01 in ProDiscover Basic⁷ browsed to \DCIM\111__06 and made a copy of IMG_0126.JPG.

Comparison of IMG_0126.JPG taken before and after the modification process shows that the photograph of the National Science Foundation has been replaced by the illustrated light bulb in the

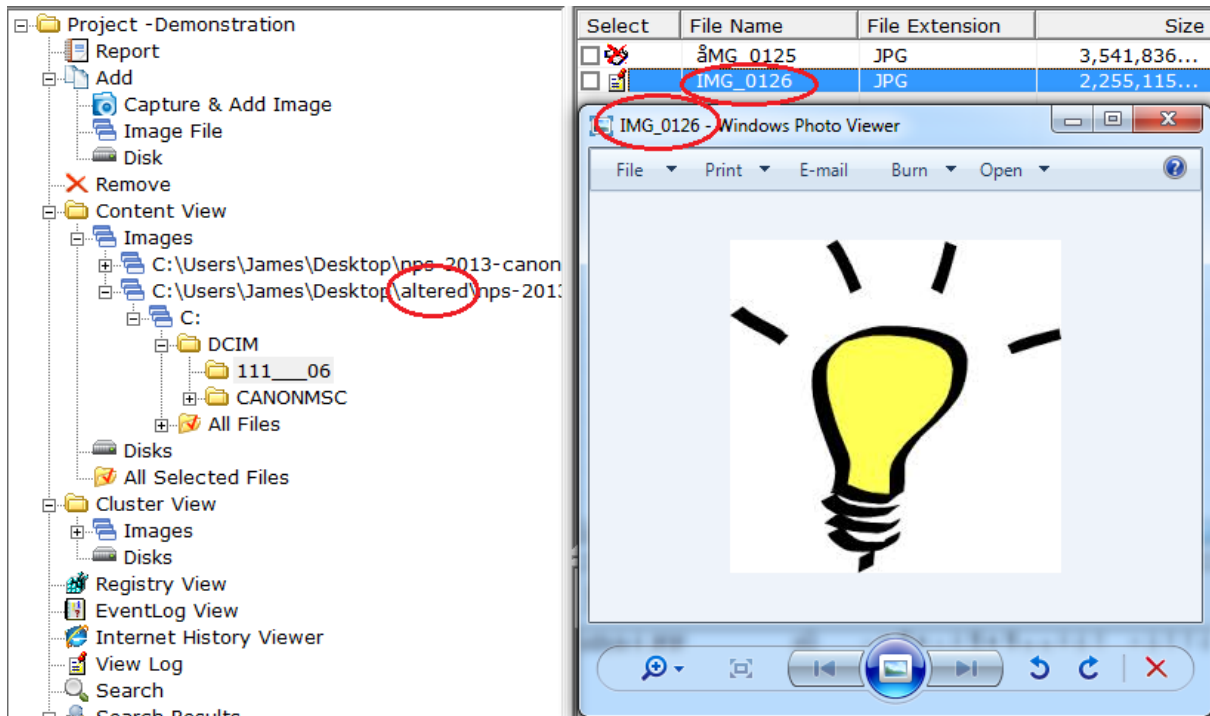


Figure 4: The contents of the IMG_0126 file after modification.

modified and reacquired digital forensic container, notwithstanding that the digital forensic containers share identical metadata. Accordingly, although the original digital evidence container has not been modified directly, a copy has been produced that shares identical characteristics to the original, except for the modifications deliberately introduced, and constitutes a meaningful modification to the digital evidence container. Therefore, H₂ is not supported.

⁷ WinHex cannot open E01 files that are not multiples of 512 bytes, despite them being considered valid by Encase Forensic Imager.

4.5 Testing of H³

H₃: Alterations to Encase Evidence File digital evidence containers cannot be detected by Encase Imager.

This hypothesis (H₃) is designed to determine whether alterations to the metadata and forensic image embedded with an Encase Evidence File digital evidence container can be detected, using Encase Forensic Imager's built in error checking and verification functions. This function is provided within the software as a single process, which verifies that metadata hashes are consistent with the forensic image and that error-checking checksums calculated for blocks of sectors have not changed. These hashes and checksums are compared between the values stored during acquisition and those generated during the verification process. Should the data have changed the values calculated during the verification process will differ from those stored during the acquisition process and indicate to the practitioner that the data in question has been altered.

Testing this hypothesis required answering two questions:

Will altering the metadata cause the validation algorithms to indicate an alteration occurred?

Will altering the embedded forensic image cause the validation algorithms to indicate an alteration occurred?

These two questions can be answered using the processes used to test H₁ and H₂ and then running the Encase Forensic Imager validation function on the resulting E01 files, one with the acquisition date altered (from H₁), and the other with the digital photograph replaced with an illustrated light bulb (from H₂).

The results of running the verification function on these altered digital forensic containers showed that the alteration to the acquisition date was not detected, suggesting that it would be difficult to refute a claim that the acquisition date had been modified.

Name	
S Module Serial Number	
S Module Firmware Version	
S Write Blocked	
S Label	
S Serial Number	
S Model	
b Read File System	•
b Parse Link Files	•
e Drive Type	Fixed
e Open Mode	Evidence File
e Source Type	Evidence File
Actual Date	01/01/14 12:00:00 AM
Target Date	01/01/14 12:00:00 AM
S Case Number	
S Examiner Name	
S Evidence Number	
S Notes	
S File Integrity	Completely Verified, 0 Errors
Acquisition MD5	7e99923ca562ad5afd9e8652621f29ca
Verification MD5	7e99923ca562ad5afd9e8652621f29ca

Figure 5: The E01 with modified acquisition dates showing no verification errors.

However, the alterations to the forensic image were easily detected, with Encase Forensic Imager providing the original hash and the new hash, and indicating the exact clusters in which alterations had been made, as indicated by changes in the checksums recorded for those sectors at acquisition time.

Accordingly, H₃ is unsupported, with the caveat that alterations to metadata are not detected.

	Name	Primary Path
<input type="checkbox"/> 1	nps-2013-canon1	C:\Users\James\Desktop\alteredimage

Fields	
Name	
S Write Blocked	
S Label	
S Serial Number	
S Model	
b Read File System	•
b Parse Link Files	•
Drive Type	Fixed
Open Mode	Evidence File
Source Type	Evidence File
Actual Date	06/03/13 05:40:06 AM
Target Date	06/03/13 05:40:06 AM
S Case Number	
S Examiner Name	
S Evidence Number	
S Notes	
S File Integrity	Completely Verified, 1 Errors
Acquisition MD5	7e99923ca562ad5afd9e8652621f29ca
Verification MD5	0eadc8e582f8d447dd3e470d0fb3de2b
Acquisition SHA1	

Figure 6: The original E01 with altered forensic image data showing a verification error.

4.6 Testing of H₄

H₄: Given two Encase Evidence File digital evidence containers with identical metadata, except the forensic image hashes, but differing forensic images, it is possible to determine which container is an accurate representation of the originally acquired device.

This hypothesis (H₄) evolved as a response to the finding of the H₃ experiment. Given that H₃ was refuted based on the hashes and checksums being recalculated and compared to previously stored values for the forensic image, but critically, not for the metadata, there existed a possibility that the metadata could be altered to reflect any alterations made to the forensic image. Accordingly, alterations were made to the modified libewf function that copied all original metadata to the new

digital forensic container, except for the hashes and file integrity checksums, which were replaced with newly calculated values corresponding to the altered forensic image.

Name	Primary Path
1 nps-2013-canon1	C:\Users\James\Desktop\nps-2013-canon1.E01

Name	
S Model	
b Read File System	•
b Parse Link Files	•
Drive Type	Fixed
Open Mode	Evidence File
Source Type	Evidence File
Actual Date	06/03/13 05:40:06 AM
Target Date	06/03/13 05:40:06 AM
S Case Number	
S Examiner Name	
S Evidence Number	
S Notes	
S File Integrity	Completely Verified, 0 Errors
Acquisition MD5	7e99923ca562ad5afd9e8652621f29ca
Verification MD5	7e99923ca562ad5afd9e8652621f29ca

Figure 7: The verification results of the original E01 file.

This new function was tested by creating a modified digital evidence container as per the instructions in the H₂ experiment and then validating this digital evidence container as per the H₃ experiment.

	Name	Primary Path
1	nps-2013-canon1	C:\Users\James\Desktop\altered\nps-2013-canon1.E01

Fields	
Name	
Drive Type	Fixed
Open Mode	Evidence File
Source Type	Evidence File
Actual Date	06/03/13 05:40:06 AM
Target Date	06/03/13 05:40:06 AM
Case Number	
Examiner Name	
Evidence Number	
Notes	
File Integrity	Completely Verified, 0 Errors
Acquisition MD5	28f538e425de42c172ef1eae565ebc90
Verification MD5	28f538e425de42c172ef1eae565ebc90

Figure 8: The verification results for the modified E01 file. Note that altered hash.

Contrary to the results of the H₃ experiment, Encase Forensic Imager detected no modifications to this digital evidence container, even though it had the same identifying information as the original, unmodified version obtained from Digital Corpora. Accordingly, using this modification of the libewf libraries creates two near identical Encase Evidence File digital evidence containers with different forensic data that both appear to be equally valid representations of the original device, which refutes the premise of H₄.

The prototype code used to acquire the raw image whilst maintaining the original metadata can be found in Appendix B.

4.7 Summary of results

H₁: The identifying metadata embedded with Encase Evidence File digital evidence containers cannot be meaningfully altered.

H₁ is found to be unsupported as identifying metadata, such as the date of acquisition, can be altered using the libewf open source library.

H₂: The forensic image embedded within Encase Evidence File digital evidence containers cannot be meaningfully altered.

H₂ is found to be unsupported as the forensic image embedded within an Encase Evidence File digital evidence container can be exported to a raw format, altered, acquired as an Encase Evidence File, and then have its metadata altered to match that of the original digital evidence container.

H₃: Alterations to Encase Evidence File digital evidence containers cannot be detected by Encase Imager.

H₃ is found to be unsupported as alterations to the forensic image can be detected by Encase Forensic Imager. Encase Forensic Imager indicates a change in the calculated hash for the forensic image and indicates which sectors have been modified. It should be noted that alterations to the metadata are not detected by Encase Forensic Imager.

H₄: Given two Encase Evidence File digital evidence containers with identical metadata, except the forensic image hashes, but differing forensic images, it is possible to determine which container is an accurate representation of the originally acquired device.

H₄ is unsupported as careful modification of the digital evidence container's metadata allows for alterations to the forensic image with it to be disguised through recalculating the metadata used during the digital evidence container verification process implemented in Encase Forensic Imager.

5 Discussion of findings

The findings of this research have dire implications and could potentially have wide ranging effects as they identify the potential for image and log sheet tampering to occur. The research highlights two primary ways of bringing digital evidence into disrepute: altering the metadata alone, or altering the metadata to disguise alterations to the forensic image, either of which could have substantial impacts on singular cases and on the future of digital evidence validation within legal practice.

5.1 Altering metadata

In H₁ it was determined that metadata could be simply altered with slight alterations to the libewf library to expose functions to the Linux command line. H₃ supplemented this finding by discovering that Encase Forensic Imager was unable to detect the alterations to the metadata field indicating the data of acquisition. This single field is shown to be vulnerable such that accusations of a practitioner performing an acquisition far later than the metadata and documentation suggests and simply backdating the acquisition date to a more appropriate time, such as mere hours following seizure, using a simple command line tool cannot be rebuffed without separate verification of the metadata. It could also be contended that during the proposed period between seizure and imaging of the source device, deliberate or accidental modification could have occurred whereby additional data may have been introduced, or existing data excised or modified, placing the reliability of the evidence in question.

Should all metadata fields be as simple to modify as the acquisition date, many more scenarios become available, limited only by the circumstances of the case and the creativity of the malicious actor. For example, the architecture information and serial numbers of a device could be altered such that it was impossible to prove the forensic image was of that particular device, leading to the potential inadmissibility of the digital evidence.

5.2 Altering forensic images

In H₂ it was determined that alteration of the forensic image was possible by creating an identical copy of the original digital evidence container. However, in H₃ it was discovered that this alteration was simply detected by Encase Forensic Imager. H₄ was therefore created to test if alterations to the forensic image could be detected if alterations to the metadata reflecting this change were also made and it was determined that Encase Forensic Imager could not detect these alterations on its own. The implications of this range from being able to replace in totality an image without a practitioner's knowledge and have them believe it was an accurate and uncompromised forensic image of the target device, through to producing a digital evidence container that is identical to another in all except content and hashes. In the latter situation, it would be impossible to determine which of two or more digital evidence containers the original, unmodified version was. This would allow a legal practitioner to introduce doubt as to the provenance of a particular digital evidence container and potentially diminish its admissibility or weight in court proceedings.

The overall effect of these discoveries is hard to determine. The legal profession is known to be conservative and slow to adopt new practices and therefore it is unlikely that any great changes to best practice and court proceedings will occur in the short term, despite the serious flaws uncovered by this research and the potential threat to proper functioning of the justice system. A more likely outcome is that a legal practitioner will introduce the concepts, findings and possibilities discussed in this research to a court and potentially influence the outcome due to the potential contamination or alteration of digital evidence. This may act as a catalyst towards faster adoption of improved practices in both the digital forensic field and legal proceedings.

5.3 Recommendations

Attacks altering the metadata only and altering forensic images both rely on a non-existent, poor, or potentially unreliable chain of custody records and documentation. In following best practice, the

hashes generated at acquisition time for a particular forensic image are recorded in the chain of custody register or case files such that any alteration to the image or the hashes recorded within a digital evidence container can be detected by comparing the current hash with that recorded previously.

This approach has two major shortcomings. Firstly, metadata beyond the hashes is often not recorded and it is not protected by the hashes. This means that any alteration to the metadata will not be detected unless it has been explicitly recorded somewhere other than within the digital evidence container. Secondly, even where this metadata, including hashes, has been recorded in a chain of custody register or case files, they are not immutable themselves. It could be alleged that any person with enough motivation to alter a digital evidence container may also have the means and motivation to alter the chain of custody or case files similarly.

Two simple measures exist to prevent these attacks, however. The first is to adopt a more comprehensive approach to recording metadata in the chain of custody register or case notes. By including every field available to the practitioner, it will be easier to determine where alterations have occurred, or what the original data was if changes are apparent but the provenance of two digital forensic containers are in question.

Secondly, these records should be collected as soon as practically possible after seizing evidence and they should be placed in the possession of an independent third party, such as an industry body, legal arbitrator, law enforcement body (for civil cases) or a university. This measure would mean that a set of metadata exists having been submitted at a known time and been protected without fear of alteration, outside of malicious network intrusion. This provides those presenting digital evidence with a much stronger chain of custody and supporting documentation with which to defend accusations of impropriety or mishandling of evidence. Anecdotal reports suggest some organisations are creating these impartial repositories; however, no academic references to the establishment of such a scheme have been located. The repositories are somewhat informal and are

designed to be used should the provenance of digital evidence be called into question, rather than acting as an official certifying body that acts to independently verify the metadata presented to courts as a means of asserting the validity of digital evidence. One such repository is administered by the Forensic Technology Group of the University of Western Australia, and consists of an email inbox that can be used to automatically store and retrieve acquisition metadata, effectively binding the metadata to a known timestamp, as recorded by the email server (The University of Western Australia, 2014).

However, it should be noted that this does not prevent a practitioner from altering, deliberately or inadvertently, the evidence between the time of seizure and the time of acquisition. This is a standing weakness in digital forensic process that relies on the integrity of the practitioner and for courts to hold practitioners to best practices, such as immediate acquisition after seizure, to limit the time in which alterations can occur.

5.4 Further research

Further research opportunities would include testing a selection of digital forensic acquisition tools and more deeply investigating those determined to be resistant to alteration. This would create further discussion within the industry and potentially propagate anti-tampering technology to all vendors and products. A valuable candidate for this research would be Guidance Software's newer format, Ex01, of which they encourage widespread adoption by practitioners and other vendors. As Guidance Software already hold a large majority of business in the digital forensics market, they are presently in a unique position to encourage advancements in forensic technology and improvements to current best practice. An analysis of how they have addressed these issues in the Ex01 format would be valuable to gauge the level of engagement vendors have with these issues.

Analysing the impact of digital signatures and encryption upon alteration attempts would prove useful in identifying improvement to current practices by publicising and promoting the use of existing features which may overcome the identified weaknesses in digital evidence validation.

Reverse engineering of the Encase Evidence File format or an analysis of the source code may be able to reveal more advanced methods for obfuscating changes to the metadata or forensic image. If a way to circumvent the validation process entirely by altering the image without need to alter the metadata could be created, it would potentially make validation of digital evidence exceedingly difficult unless someone was looking for specific signs of alteration.

Further general research into the processes and equipment used to acquire forensic images and generate acquisition process logs would build a larger knowledge base which legal practitioners and judges could draw upon to assess and validate digital evidence. Deeper investigation from a legal practice perspective of the methods by which practitioners present and validate digital evidence in courtrooms may discover ways to promote more robust validation practices and requirements through raising awareness in legal practitioners and the judiciary.

Furthermore, anecdotal evidence suggests that at least one vendor of digital forensic software, Perlustro L.P., (<http://www.perlustro.com/>) has for some time prevented metadata alteration attacks against its image containers and acquisition audit log. This is purportedly achieved through an embedded, encrypted logging mechanism that details all changes to the digital evidence container over its lifetime. Further research would be required to scientifically validate these claims and assess their efficacy in preventing the alteration of the metadata of their digital evidence containers.

6 Conclusion

The validation of digital evidence is an important process in presenting digital evidence in court proceedings. Validation of forensic images ensures that the evidence presented is a truthful and accurate representation of the source device at the time it was seized and allows decision makers to reach a verdict based on the truth of a matter. Therefore, any weaknesses in the validation process may allow contaminated or fraudulent evidence to be accepted by a court and used in consideration of a verdict, with the potential to affect the outcome of a case adversely.

Current practice for the validation of evidence involves the presentation of a chain of custody register, acquisition process reports and metadata, and a forensic image. The reports are generated from metadata that is embedded within a digital evidence container along with an embedded forensic image. It has been shown, however, that both the metadata and the forensic image can be altered in such a way that it may not be possible to determine an alteration has occurred, or it may not be possible to determine which of two almost identical digital evidence containers is an accurate and unaltered representation of the original data.

The implications of these findings include showing that malicious actors have the means to alter digital evidence and therefore potentially influence digital forensic practitioners, legal practitioners and ultimately the proceedings of a court. Courts and the justice system operate in such a manner where evidence must be of utmost integrity to ensure that the outcome of court cases are within the law, truth and the interests of justice.

The potential attacks against digital evidence are simply thwarted, however, using independent validation of digital evidence metadata. By retaining identifying information in the care of an disinterested and unbiased third party the court and those legal practitioners who rely on digital evidence can more easily counter claims that their evidence may be altered or otherwise unreliable.

References

- Adams, R. (2012). *The Advanced Data Acquisition Model (ADAM): A process model for digital forensic practice*. Murdoch University. Retrieved from <http://researchrepository.murdoch.edu.au/14422/>
- Akester, P. (2004). Authorship and authenticity in cyberspace. *Computer Law & Security Review*, 20(6), 436–444.
- Altheide, C., & Carvey, H. A. (2011). *Digital forensics with open source tools using open source platform tools for performing computer forensics on target systems: Windows, Mac, Linux, UNIX, etc.* Burlington, MA: Syngress.
- Aquilina, J., Casey, E., & Malin, C. (2008). *Malware forensics : investigating and analyzing malicious code*. Burlington, MA: Syngress.
- Ashcroft, J. (2001). A Guide for First Responders. *National Institute of Justice*, 4. Retrieved from <http://www.iwar.org.uk/ecoespionage/resources/cybercrime/ecrime-scene-investigation.pdf>
- Ashley, K., & Rissland, E. (1986). Toward modelling legal argument. In A. Martino & F. Natali (Eds.), *Automated analysis of legal texts: logic, informatics, law*. Florence, Italy.
- Bell, G. B., & Boddington, R. (2010). Solid state drives: the beginning of the end for current practice in digital forensic recovery? *Journal of Digital Forensics, Security & Law*, 5(3).
- Berg, E. (2000). Legal ramifications of digital imaging in law enforcement. *Forensic Science Communications*, 2(4).
- Buchholz, F., & Spafford, E. (2004). On the role of file system metadata in digital forensics. *Digital Investigation*, 1(4), 298–309.
- Caloyannides, M. (2004). *Privacy protection and computer forensics (2nd ed.)*. Norwood, MA, USA: Artech House.

- Carrier, B. (2003). Defining digital forensic examination and analysis tools using abstraction layers. *International Journal of Digital Evidence*, 1(4), 1–12.
- Carrier, B. (2005). *File system forensic analysis*. Boston, Mass.; London: Addison-Wesley.
- Carrier, B. D. (2006). Risks of live digital forensic analysis. *Communications of the ACM*, 49(2), 56–61.
- Carrier, B., & Spafford, E. H. (2003). Getting physical with the digital investigation process. *International Journal of Digital Evidence*, 2(2), 1–20.
- Casey, E. (2011). *Digital evidence and computer crime: forensic science, computers and the Internet*. Waltham, MA: Academic Press.
- CDESWFG. (2006). *Survey of disk image storage formats*. Common Digital Evidence Storage Format Working Group, Digital Forensic Research Workshop.
- Cohen, F. (2008). *Challenges to digital forensic evidence*. Fred Cohen and Associates.
- Corey, V., Peterman, C., Shearin, S., Greenberg, M. S., & Van Bokkelen, J. (2002). Network forensics analysis. *Internet Computing, IEEE*, 6(6), 60–66.
- Cosic, J., & Baca, M. (2010). Do we have full control over integrity in digital evidence life cycle? In *Information Technology Interfaces (ITI), 2010 32nd International Conference on* (pp. 429–434). IEEE. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5546413
- Craiger, P., Swauger, J., Marberry, C., & Hendricks, C. (2006). Validation of digital forensics tools. In P. Kanellis, E. Kiountouzis, N. Kolokotronis, & D. Martakos (Eds.), *Digital Crime and Forensic Science in Cyberspace* (p. 91). Hershey, PA: Idea Group Publishing.
- De Maio, G. (2013). *On the evolution of digital evidence: novel approaches for cyber investigation*. University of Salerno. Retrieved from <http://elea.unisa.it:8080/handle/10556/1443>
- Digital Forensics Research Workshop. (2001). *A Road Map for Digital Forensic Research*.
- Draft Guidance: Digital Forensics Method Validation. (2014). Forensic Science Regulator.
- Duerr, T. E., Beser, N. D., & Stasiunas, G. P. (2004). Information assurance applied to authentication of digital evidence. *Forensic Science Communications*, 6(4).

- Etter, B. (2001). Computer crime. In *4th National Outlook Symposium on Crime in Australia-New Crimes or New Responses*. Retrieved from http://www.aic.gov.au/media_library/conferences/outlook4/etter.pdf
- Federal Rules of Evidence. (2010). U.S. Government Printing Office. Retrieved from <http://www.uscourts.gov/uscourts/rulesandpolicies/rules/2010%20rules/evidence.pdf>
- Garfinkel, S., Farrell, P., Rousev, V., & Dinolt, G. (2009). Bringing science to digital forensics with standardized forensic corpora. *Digital Investigation*, 6, S2–S11.
- Garfinkel, S. L. (2010). Digital forensics research: The next 10 years. *Digital Investigation*, 7, S64–S73.
- Gottesman, M. H. (1994). Admissibility of expert testimony after Daubert: The prestige factor. *Emory LJ*, 43(3), 867.
- Guo, Y., Slay, J., & Beckett, J. (2009). Validation and verification of computer forensic software tools—Searching Function. *Digital Investigation*, 6, S12–S22.
- Gupta, M. R., Hoeschele, M. D., & Rogers, M. K. (2006). Hidden disk areas: HPA and DCO. *International Journal of Digital Evidence*, 5(1), 1–8.
- Hardware Write Blocker Device (HWB) Specification Version 2.0. (2004). NIST. Retrieved from <https://www.utica.edu/academic/institutes/ecii/publications/articles/A04BC142-F4C3-EB2B-462CCC0C887B3CBE.pdf>
- High Court of Australia. (2000). *Osland v R* [1998] HCA 75; 197 CLR 316; 159 ALR 170; 73 ALJR 173 (10 December 1998). Australasian Legal Information Institute.
- Hosmer, C. (2002). Proving the Integrity of Digital Evidence with Time. *International Journal of Digital Evidence*, 1(1).
- Jarrett, H. M., Bailie, M. W., Hagen, E., & Judish, N. (2009). Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations. Office of Legal Education, Executive Office for United States Attorneys.

- Koehler, J. J., & Thompson, W. C. (2006). Mock jurors' reactions to selective presentation of evidence from multiple-opportunity searches. *Law and Human Behavior, 30*(4), 455–468.
doi:10.1007/s10979-006-9021-4
- Lenstra, A., & de Wegner, B. (2005). On the Possibility of Constructing Meaningful Hash Collisions for Public Keys. In C. Boyd & J. Nieto (Eds.), *Information security and privacy: 10th Australasian conference ; proceedings* (pp. 267–279). Berlin: Springer.
- Lyle, J. R. (2003). NIST CFTT: Testing disk imaging tools. *International Journal of Digital Evidence, 1*(4), 1–10.
- Lyle, J. R., & Black, P. E. (2005). *Testing BIOS Interrupt 0x13 Based Software Write Blockers*. National Institute of Standards and Technology. Retrieved from
http://xmlregistry.nist.gov/~black/Papers/testSWB_ECCE05.txt
- Lyle, J. R., & Wozar, M. (2007). Issues with imaging drives containing faulty sectors. *Digital Investigation, 4*, 13–15. doi:10.1016/j.diin.2007.06.002
- Mason, S. (2007). *Electronic evidence: disclosure, discovery and admissibility*. London: LexisNexis Butterworths.
- Mercuri, R. (2005). Challenges in forensic computing. *Communications of the ACM, 48*(12), 17–21.
- Mocas, S. (2004). Building theoretical underpinnings for digital forensics research. *Digital Investigation, 1*(1), 61–68. doi:10.1016/j.diin.2003.12.004
- Moles, R. N. (2007). The role and function of the expert witness.
- Mouhtaropoulos, A., Li, C.-T., & Grobler, M. (2014). Digital forensic readiness: are we there yet? *Journal of International Commercial Law and Technology, 9*(3).
- Nfuka, E. N., Sanga, C., & Mshangi, M. (2014). The rapid growth of cybercrimes affecting information systems in the global: Is this a myth or reality in Tanzania? *International Journal of Information Security Science, 3*(2), 182–199.
- Noblett, M. G., Pollitt, M. M., & Presley, L. A. (2000). Recovering and examining computer forensic evidence. *Forensic Science Communications, 2*(4).

- Oppliger, R., & Rytz, R. (2003). Digital evidence: Dream and reality. *IEEE Security & Privacy Magazine*, 1(5), 44–48.
- Perelman, C., Weaver, P., Wilkinson, J., & Olbrechts-Tyteca, L. (1971). *The new rhetoric: A treatise on argumentation*. London: University of Notre Dame Press.
- Pollitt, M. M. (2001). Report on digital evidence. In *13th INTERPOL Forensic Science Symposium*. Retrieved from <https://www.interpol.int/Public/Forensic/IFSS/meeting13/Reviews/Digital.pdf>
- Richard, G. G., Roussev, V., & Marziale, L. (2007). Forensic discovery auditing of digital evidence containers. *Digital Investigation*, 4(2), 88–97. doi:10.1016/j.diin.2007.04.002
- Rogers, M., Scarborough, K., Frakes, K., & San Martin, C. (2007). Survey of law enforcement perceptions regarding digital evidence. In P. Craiger & S. Sheno (Eds.), *Advances in Digital Forensics III* (pp. 41–52). Springer.
- Roussev, V., Chen, Y., Bourg, T., & Richard, G. G. (2006). md5bloom: Forensic filesystem hashing revisited. *Digital Investigation*, 3, 82–90. doi:10.1016/j.diin.2006.06.012
- Rowlingson, R. (2004). A ten step process for forensic readiness. *International Journal of Digital Evidence*, 2(3), 1–28.
- Ruff, N. (2008). Windows memory forensics. *Journal in Computer Virology*, 4(2), 83–100. doi:10.1007/s11416-007-0070-0
- Saferstein, R. (Ed.). (2001). *Forensic science handbook*. Englewood Cliffs, N.J.: Prentice Hall.
- Saferstein, R. (2011). *Criminalistics : an introduction to forensic science* (10th ed.). Boston, MA: Pearson Education.
- Schneier, B. (2004). Opinion: Cryptanalysis of MD5 and SHA: Time for a new standard: Crypto researchers report weaknesses in common hash functions. *Computerworld*.
- Silverstone, H., & Sheetz, M. (2007). *Forensic accounting and fraud investigation for non-experts* (2nd ed.). New Jersey: John Wiley & Sons.
- Software Write Block Tool Specification & Test Plan Version 3.0. (2003). NIST.

- Sommer, P. (1998). Digital footprints: Assessing computer evidence. *Criminal Law Review*, 12, 61–78.
- The University of Western Australia. (2014). Forensic Technology Group. Retrieved November 16, 2014, from <http://www.forensicscience.uwa.edu.au/research/forensic-technology-group>
- Thompson, D. E., & Berwick, D. (1998). *Minimum provisions for the investigation of computer based offences*. Payneham, SA: National Police Research Unit.
- Turner, P. (2005). Digital provenance – interpretation, verification and corroboration. *Digital Investigation*, 2(1), 45–49. doi:10.1016/j.diin.2005.01.002
- Turner, P. (2006). Selective and intelligent imaging using digital evidence bags. *Digital Investigation*, 3, 59–64. doi:10.1016/j.diin.2006.06.003
- Wang, X., & Yu, H. (2006). How to break MD5 and other hash functions. In R. Cramer (Ed.), *Advances in Cryptology*. Guildford: Springer London.
- Windows support for hard disks that are larger than 2 TB. (2013). Microsoft Corporation. Retrieved from <http://support.microsoft.com/kb/2581408>


```

    libcstring_system_character_t *option_header_codepage           = NULL;
+   libcstring_system_character_t *option_acquiry_date = NULL;
    libcstring_system_character_t *option_maximum_segment_size     = NULL;
    libcstring_system_character_t *option_offset                   = NULL;
    libcstring_system_character_t *option_process_buffer_size     = NULL;
@@ -321,7 +324,7 @@ int main( int argc, char * const argv[] )
    while( ( option = libcsystem_getopt(
        argc,
        argv,
-       _LIBCSTRING_SYSTEM_STRING(
"A:b:B:c:d:f:hl:o:p:qsS:t:uvVwx" ) ) ) != (libcstring_system_integer_t) -1
)
+       _LIBCSTRING_SYSTEM_STRING(
"A:b:B:c:d:f:hl:m:o:p:qsS:t:uvVwx" ) ) ) != (libcstring_system_integer_t) -
1 )
    {
        switch( option )
        {
@@ -386,6 +389,11 @@ int main( int argc, char * const argv[] )

            break;

+           case (libcstring_system_integer_t) 'm':
+               option_acquiry_date = optarg;
+               break;
+
            case (libcstring_system_integer_t) 'o':
                option_offset = optarg;

@@ -714,6 +722,29 @@ int main( int argc, char * const argv[] )
                ewfexport_export_handle->sectors_per_chunk );
        }
    }
+   if (option_acquiry_date != NULL)
+   {
+       if (export_handle_set_string(
+           ewfexport_export_handle,
+           option_acquiry_date,
+           &(ewfexport_export_handle->acquiry_date),
+           &(ewfexport_export_handle->acquiry_date_size),
+           &error) != 1)
+       {
+           fprintf(
+               stderr,
+               "Unable to set acquiry date.\n");
+           goto on_error;
+       }
+   }
+   else{
+       fprintf(
+           stderr,
+           "In this version of ewfexport, you must specify an
acquiry date (-m).\n");
+       goto on_error;
+   }
    if( option_maximum_segment_size != NULL )
    {
        result = export_handle_set_maximum_segment_size(

```

```

diff --git a/ewftools/export_handle.c b/ewftools/export_handle.c
index 95e30a7..7d4dbb3 100644
--- a/ewftools/export_handle.c
+++ b/ewftools/export_handle.c
@@ -3712,6 +3712,7 @@ int export_handle_set_output_values(
    if( libewf_handle_copy_header_values(
        export_handle->ewf_output_handle,
        export_handle->input_handle,
+       (uint8_t *) export_handle->acquiry_date,
        error ) != 1 )
    {
        libcerror_error_set(
@@ -3723,8 +3724,9 @@ int export_handle_set_output_values(
        return( -1 );
    }
-   /* Set acquiry operating system, software and software version
+   /* Set acquiry operating system, software and software version
from source file
   */
+   /*
    if( acquiry_operating_system != NULL )
    {
        value_string_length = libcstring_system_string_length(
@@ -3821,6 +3823,7 @@ int export_handle_set_output_values(
        return( -1 );
    }
+   */
    if( libewf_handle_set_header_codepage(
        export_handle->ewf_output_handle,
        export_handle->header_codepage,
diff --git a/ewftools/export_handle.h b/ewftools/export_handle.h
index fc09ebc..a3673fa 100644
--- a/ewftools/export_handle.h
+++ b/ewftools/export_handle.h
@@ -97,6 +97,14 @@ struct export_handle
    */
    uint64_t export_size;

+   /* The acquiry date
+   */
+   libcstring_system_character_t *acquiry_date;
+
+   /* The acquiry date size
+   */
+   uint64_t acquiry_date_size;
+
    /* The header codepage
    */
    int header_codepage;
diff --git a/include/libewf.h b/include/libewf.h
index 1b46c49..058c5ff 100644
--- a/include/libewf.h
+++ b/include/libewf.h
@@ -1608,6 +1608,7 @@ LIBEWF_EXTERN \
int libewf_handle_copy_header_values(
    libewf_handle_t *destination_handle,
    libewf_handle_t *source_handle,
+   uint8_t *acquiry_date,
    libewf_error_t **error );

```

```

/* Retrieves the number of hash values
diff --git a/include/libewf.h.in b/include/libewf.h.in
index 31d7870..48f63ee 100644
--- a/include/libewf.h.in
+++ b/include/libewf.h.in
@@ -1608,6 +1608,7 @@ LIBEWF_EXTERN \
int libewf_handle_copy_header_values(
    libewf_handle_t *destination_handle,
    libewf_handle_t *source_handle,
+   uint8_t *acquiry_date,
    libewf_error_t **error );

/* Retrieves the number of hash values
diff --git a/libewf/libewf_header_values.c b/libewf/libewf_header_values.c
index 125536f..c87b089 100644
--- a/libewf/libewf_header_values.c
+++ b/libewf/libewf_header_values.c
@@ -1308,6 +1308,7 @@ on_error:
int libewf_header_values_copy(
    libfvalue_table_t *destination_header_values,
    libfvalue_table_t *source_header_values,
+   uint8_t *acquiry_date,
    libcerror_error_t **error )
{
    libfvalue_value_t *destination_header_value = NULL;
@@ -1399,22 +1400,22 @@ int libewf_header_values_copy(
    /* Ignore the acquiry and system date
    * They will be auto generated
    */
-   else if( ( identifier_size == 13 )
+   else if(
+       (( identifier_size == 13 )
+        && ( libcstring_narrow_string_compare(
+            (char *) identifier,
+            "acquiry_date",
+            12 ) == 0 ) )
+       ((identifier_size == 12)
+        && (libcstring_narrow_string_compare(
+            (char *) identifier,
+            "system_date",
+            11) == 0))
+   )
    {
        continue;
    }
-   else if( ( identifier_size == 12 )
-       && ( libcstring_narrow_string_compare(
-           (char *) identifier,
-           "system_date",
-           11 ) == 0 ) )
    {
        continue;
+       libfvalue_value_set_data(source_header_value,
acquiry_date, 20, LIBFVALUE_CODEPAGE_UTF8,
LIBFVALUE_VALUE_FLAG_DATA_MANAGED, error);
    }
+
    /* Ignore empty values
    */

```

```

        result = libfvalue_value_has_data(
diff --git a/libewf/libewf_header_values.h b/libewf/libewf_header_values.h
index 2768ae9..c87ceee 100644
--- a/libewf/libewf_header_values.h
+++ b/libewf/libewf_header_values.h
@@ -143,6 +143,7 @@ int libewf_generate_date_header2_value(
    int libewf_header_values_copy(
        libfvalue_table_t *destination_header_values,
        libfvalue_table_t *source_header_values,
+    uint8_t *acquiry_date,
        libcerror_error_t **error );

    int libewf_header_values_parse_utf8_header_string(
diff --git a/libewf/libewf_metadata.c b/libewf/libewf_metadata.c
index ec320e6..321e3cd 100644
--- a/libewf/libewf_metadata.c
+++ b/libewf/libewf_metadata.c
@@ -4466,6 +4466,7 @@ int libewf_handle_set_utf16_header_value(
    int libewf_handle_copy_header_values(
        libewf_handle_t *destination_handle,
        libewf_handle_t *source_handle,
+    uint8_t *acquiry_date,
        libcerror_error_t **error )
    {
        libewf_internal_handle_t *internal_destination_handle = NULL;
@@ -4527,6 +4528,7 @@ int libewf_handle_copy_header_values(
    if( libewf_header_values_copy(
        internal_destination_handle->header_values,
        internal_source_handle->header_values,
+    acquiry_date,
        error ) != 1 )
    {
        libcerror_error_set(
diff --git a/libewf/libewf_metadata.h b/libewf/libewf_metadata.h
index 443c378..cffe9a 100644
--- a/libewf/libewf_metadata.h
+++ b/libewf/libewf_metadata.h
@@ -401,6 +401,7 @@ LIBEWF_EXTERN \
    int libewf_handle_copy_header_values(
        libewf_handle_t *destination_handle,
        libewf_handle_t *source_handle,
+    uint8_t *acquiry_date,
        libcerror_error_t **error );

LIBEWF_EXTERN \

```

Appendix B

The following is the result of a diff operation between the distributed libewf library and the prototype code developed for testing H₄. Compiling required the zlib1g-dev package on Kali Linux.

```
ewftools/ewfacquire.c      | 23 ++++-
ewftools/imaging_handle.c  | 189
+++++++-----
ewftools/imaging_handle.h  | 8 ++
libewf/libewf_header_values.c | 21 +----
4 files changed, 210 insertions(+), 31 deletions(-)

diff --git a/ewftools/ewfacquire.c b/ewftools/ewfacquire.c
index 51bc71a..8927c0a 100644
--- a/ewftools/ewfacquire.c
+++ b/ewftools/ewfacquire.c
@@ -113,7 +113,7 @@ void ewfacquire_usage_fprint(
    "                    [ -B number_of_bytes ] [ -c
compression_values ]\n"
    "                    [ -C case_number ] [ -d
digest_type ] [ -D description ]\n"
    "                    [ -e examiner_name ] [ -E
evidence_number ] [ -f format ]\n"
-   "                    [ -g number_of_sectors ] [ -l
log_filename ]\n"
+   "                    [ -g number_of_sectors ]
[-9 filename to source headers] [ -l log_filename ]\n"
    "                    [ -m media_type ] [ -M
media_flags ] [ -N notes ]\n"
    "                    [ -o offset ] [ -p
process_buffer_size ]\n"
    "                    [ -P bytes_per_sector ] [ -r
read_error_retries ]\n"
@@ -135,6 +135,7 @@ int main( int argc, char * const argv[] )
    libcerrror_error_t *error =
NULL;
+   libcstring_system_character_t *option_source_headers_filename =
NULL;
    libcstring_system_character_t *log_filename =
NULL;
    libcstring_system_character_t *option_additional_digest_types =
NULL;
    libcstring_system_character_t *option_bytes_per_sector =
NULL;
@@ -162,6 +162,11 @@ int main( int argc, char * const argv[] )
    option_secondary_target_filename = optarg;

    break;

+   case (libcstring_system_integer_t) '9':
+       option_source_headers_filename = optarg;
+       break;
    }
}
```

```

        if( optind == argc )
@@ -1913,6 +1919,21 @@ int main( int argc, char * const argv[] )
            goto on_error;
        }
    }
+   if (option_source_headers_filename != NULL)
+   {
+       if (imaging_handle_set_string(
+           ewfacquire_imaging_handle,
+           option_source_headers_filename,
+           &(ewfacquire_imaging_handle->source_headers_filename),
+           &(ewfacquire_imaging_handle-
>source_headers_filename_size),
+           &error) != 1){
+
+           fprintf(stderr, "Unable to set source headers filename.\n
");
+
+           goto on_error;
+       }
+   }
+
    if( option_examiner_name != NULL )
    {
        if( imaging_handle_set_string(
diff --git a/ewftools/imaging_handle.c b/ewftools/imaging_handle.c
index 285a258..9bd2a4e 100644
--- a/ewftools/imaging_handle.c
+++ b/ewftools/imaging_handle.c
@@ -50,6 +50,150 @@
#define IMAGING_HANDLE_STRING_SIZE          1024
#define IMAGING_HANDLE_NOTIFY_STREAM       stdout

+
+
+/* Opens the input of the export handle
+* Returns 1 if successful or -1 on error
+*/
+int export_handle_open_input(
+    imaging_handle_t *export_handle,
+    libcstring_system_character_t * const * filenames,
+    int number_of_filenames,
+    libcerror_error_t **error)
+{
+
+
+    libcstring_system_character_t **libewf_filenames = NULL;
+    static char *function = "export_handle_open_input";
+    size_t first_filename_length = 0;
+
+    if (number_of_filenames <= 0)
+    {
+        libcerror_error_set(
+            error,
+            LIBCERROR_ERROR_DOMAIN_ARGUMENTS,
+            LIBCERROR_ARGUMENT_ERROR_VALUE_ZERO_OR_LESS,
+            "%s: invalid number of filenames.",
+            function);
+
+        return(-1);
+    }

```



```

+     if (number_of_filenames == 1)
+     {
+         first_filename_length = libcstring_system_string_length(
+             filenames[0]);
+
+ #if defined( LIBCSTRING_HAVE_WIDE_SYSTEM_CHARACTER )
+     if (libewf_glob_wide(
+         filenames[0],
+         first_filename_length,
+         LIBEWF_FORMAT_UNKNOWN,
+         &libewf_filenames,
+         &number_of_filenames,
+         error) != 1)
+ #else
+     if (libewf_glob(
+         filenames[0],
+         first_filename_length,
+         LIBEWF_FORMAT_UNKNOWN,
+         &libewf_filenames,
+         &number_of_filenames,
+         error) != 1)
+ #endif
+     {
+         libcerror_error_set(
+             error,
+             LIBCERROR_ERROR_DOMAIN_RUNTIME,
+             LIBCERROR_RUNTIME_ERROR_GET_FAILED,
+             "%s: unable to resolve filename(s).",
+             function);
+
+         return(-1);
+     }
+     filenames = (libcstring_system_character_t * const *)
libewf_filenames;
+ }
+
+ if (libewf_handle_initialize(
+     &( export_handle->headers_source_handle),
+     error) != 1)
+ {
+     libcerror_error_set(
+         error,
+         LIBCERROR_ERROR_DOMAIN_RUNTIME,
+         LIBCERROR_RUNTIME_ERROR_INITIALIZE_FAILED,
+         "%s: unable to create input handle.",
+         function);
+
+     return (-1);
+ }
+
+ #if defined( LIBCSTRING_HAVE_WIDE_SYSTEM_CHARACTER )
+     if (libewf_handle_open_wide(
+         export_handle->headers_source_handle,
+         filenames,
+         number_of_filenames,
+         LIBEWF_OPEN_READ,
+         error) != 1)
+ #else
+     if (libewf_handle_open(
+         export_handle->headers_source_handle,
+         filenames,

```

```

+         number_of_filenames,
+         LIBEWF_OPEN_READ,
+         error) != 1)
+ #endif
+     {
+         libcerror_error_set(
+             error,
+             LIBCERROR_ERROR_DOMAIN_IO,
+             LIBCERROR_IO_ERROR_OPEN_FAILED,
+             "%s: unable to open file(s).",
+             function);
+
+         if (libewf_filenames != NULL)
+         {
+ #if defined( LIBCSTRING_HAVE_WIDE_SYSTEM_CHARACTER )
+             libewf_glob_free(
+                 libewf_filenames,
+                 number_of_filenames,
+                 NULL);
+ #else
+             libewf_glob_free(
+                 libewf_filenames,
+                 number_of_filenames,
+                 NULL);
+ #endif
+         }
+         return(-1);
+     }
+     if (libewf_filenames != NULL)
+     {
+ #if defined( LIBCSTRING_HAVE_WIDE_SYSTEM_CHARACTER )
+         if (libewf_glob_free(
+             libewf_filenames,
+             number_of_filenames,
+             error) != 1)
+ #else
+         if (libewf_glob_free(
+             libewf_filenames,
+             number_of_filenames,
+             error) != 1)
+ #endif
+         {
+             libcerror_error_set(
+                 error,
+                 LIBCERROR_ERROR_DOMAIN_RUNTIME,
+                 LIBCERROR_RUNTIME_ERROR_FINALIZE_FAILED,
+                 "%s: unable to free globbed filenames.",
+                 function);
+
+             return(-1);
+         }
+     }
+     return(1);
+ }
+
+ /* Creates an imaging handle
+  * Make sure the value imaging_handle is referencing, is set to NULL
+  * Returns 1 if successful or -1 on error
+ @@ -4198,7 +4342,7 @@ int imaging_handle_set_output_values(

```

```

        return( -1 );
    }
-   if( imaging_handle->case_number != NULL )
+   if ( imaging_handle->source_headers_filename != NULL &&
imaging_handle->case_number != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4217,7 +4361,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        )
    }
-   if( imaging_handle->description != NULL )
+   if (imaging_handle->source_headers_filename != NULL &&
imaging_handle->description != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4236,7 +4380,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        )
    }
-   if( imaging_handle->evidence_number != NULL )
+   if (imaging_handle->source_headers_filename != NULL &&
imaging_handle->evidence_number != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4255,7 +4399,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        )
    }
-   if( imaging_handle->examiner_name != NULL )
+   if (imaging_handle->source_headers_filename != NULL &&
imaging_handle->examiner_name != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4274,7 +4418,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        )
    }
-   if( imaging_handle->notes != NULL )
+   if (imaging_handle->source_headers_filename != NULL &&
imaging_handle->notes != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4301,7 +4445,7 @@ int imaging_handle_set_output_values(

        /* Set acquiry operating system, software and software version
        */
-   if( platform_get_operating_system(
+   if (imaging_handle->source_headers_filename != NULL &&
platform_get_operating_system(
        acquiry_operating_system,
        32,
        error ) != 1 )
@@ -4324,7 +4468,7 @@ int imaging_handle_set_output_values(
        libcerror_error_free(
            error );
    }

```

```

    }
-   else
+   else if (imaging_handle->source_headers_filename != NULL )
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4343,7 +4487,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        }
    }
-   if( acquiry_software != NULL )
+   if ( imaging_handle->source_headers_filename != NULL &&
acquiry_software != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4362,7 +4506,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        }
    }
-   if( acquiry_software_version != NULL )
+   if (imaging_handle->source_headers_filename != NULL &&
acquiry_software_version != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4381,7 +4525,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        }
    }
-   if( model != NULL )
+   if (imaging_handle->source_headers_filename != NULL && model != NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4400,7 +4544,7 @@ int imaging_handle_set_output_values(
            return( -1 );
        }
    }
-   if( serial_number != NULL )
+   if (imaging_handle->source_headers_filename != NULL && serial_number
!= NULL)
    {
        if( imaging_handle_set_header_value(
            imaging_handle,
@@ -4491,6 +4635,29 @@ int imaging_handle_set_output_values(
    }
    /* Format needs to be set before segment file size and compression
values
*/
+
+   if (imaging_handle->source_headers_filename != NULL){
+
+       libcstring_system_character_t * files[1] = { imaging_handle-
>source_headers_filename };
+
+       if (export_handle_open_input(imaging_handle, files, 1, error)
!= 1){
+           fprintf(stderr, "Error opening source headers file.");
+           return (-1);
+       }
    }

```

```

+
+         if (libewf_handle_copy_header_values(
+             imaging_handle->output_handle,
+             imaging_handle->headers_source_handle,
+             error) != 1)
+         {
+             fprintf(
+                 stderr,
+                 "unable to copy header values.");
+
+             return(-1);
+         }
+     }
+
+     if( libewf_handle_set_format(
+         imaging_handle->output_handle,
+         imaging_handle->ewf_format,
diff --git a/ewftools/imaging_handle.h b/ewftools/imaging_handle.h
index 4ec946a..4ee0fe3 100644
--- a/ewftools/imaging_handle.h
+++ b/ewftools/imaging_handle.h
@@ -52,6 +52,11 @@ struct imaging_handle
    */
    size_t target_filename_size;

+   /* Filename for source headers */
+   libcstring_system_character_t *source_headers_filename;
+
+   size_t source_headers_filename_size;
+
    /* The secondary target filename
    */
    libcstring_system_character_t *secondary_target_filename;
@@ -212,6 +217,9 @@ struct imaging_handle
    */
    libewf_handle_t *secondary_output_handle;

+   /* Source handle for copying headers */
+   libewf_handle_t *headers_source_handle;
+
    /* The input media size
    */
    size64_t input_media_size;
diff --git a/libewf/libewf_header_values.c b/libewf/libewf_header_values.c
index 125536f..32da143 100644
--- a/libewf/libewf_header_values.c
+++ b/libewf/libewf_header_values.c
@@ -1396,25 +1396,7 @@ int libewf_header_values_copy(
    #endif

        continue;
    }

-   /* Ignore the acquiry and system date
-   * They will be auto generated
-   */
-   else if( ( identifier_size == 13 )
-            && ( libcstring_narrow_string_compare(
-                (char *) identifier,
-                "acquiry_date",
-                12 ) == 0 ) )
-   {
        continue;
    }

```

```

-         }
-     else if( ( identifier_size == 12 )
-             && ( libcstring_narrow_string_compare(
-                 (char *) identifier,
-                 "system_date",
-                 11 ) == 0 ) )
-     {
-         continue;
-     }
+
+     /* Ignore empty values
+      */
+     result = libfvalue_value_has_data(
@@ -1437,6 +1419,7 @@ int libewf_header_values_copy(
+     {
+         continue;
+     }
+
+     if( libfvalue_value_clone(
+         &destination_header_value,
+         source_header_value,

```