Forensics Knowledge Area Version 1.0.1

Vassil Roussev | University of New Orleans

EDITOR
Howard Chivers | University of York

REVIEWERS
Bruce Nikkel | Bern University of Applied Sciences
Marc Kirby | Oracle
Paul Birch | BDO
Zeno Geradts | University of Amsterdam

COPYRIGHT

CHANGELOG

Version date	Version number	Changes made
July 2021	1.0.1	Updated copyright statement; amended “issue” to “version”
October 2019	1.0	

INTRODUCTION

Digital forensic science, or *digital forensics*, is the application of scientific tools and methods to identify, collect and analyse digital (data) artifacts in support of legal proceedings. From a technical perspective, it is the process of identifying and reconstructing the *relevant* sequence of events that has led to the currently observable state of a target IT system or (digital) artifacts. The importance of digital evidence has grown in lockstep with the fast societal adoption of information technology, which has resulted in the continuous accumulation of data at an exponential rate. Simultaneously, there has been rapid growth in network connectivity and the complexity of IT systems, leading to more complex behaviour that may need investigation.

The primary purpose of this Knowledge Area is to provide a technical overview of digital forensic techniques and capabilities, and to put them into a broader perspective with regard to other related areas in the cybersecurity domain. The discussion on legal aspects of digital forensics is limited only to general principles and best practices, as the specifics of the application of these principles tend to vary across jurisdictions. For example, the Knowledge Area discusses the availability of different types of evidence, but does not work through the legal processes that have to be followed to obtain them. The Law & Regulation CyBOK Knowledge Area [1] discusses specific concerns related to jurisdiction and the legal process to obtain, process, and present digital evidence.

CONTENT

1 DEFINITIONS AND CONCEPTUAL MODELS

[2, 3, 4, 5, 6]

Broadly, *forensic science* is the application of scientific methods to collect, preserve and analyse evidence related to legal cases [2]. Historically, this involved the systematic analysis of (samples of) physical material in order to establish causal relationships between various events, as well as to address issues of provenance and authenticity. The rationale behind it, *Locard's exchange principle*, is that physical contact between objects inevitably results in the exchange of matter, leaving *traces* that can be analysed to (partially) reconstruct the event.

With the introduction of digital computing and communication, which we refer to as the *cyber domain*, the same general assumptions were extended largely unchallenged. Although a detailed conceptual discussion is beyond the scope of this chapter, it is important to recognise that the presence of a persistent *digital (forensic) trace* is neither inevitable, nor is it a "natural" consequence of the processing and communication of digital information.

A *digital (forensic) trace* is an explicit, or implicit, record that testifies to the execution of specific computations, or the communication and/or storage of specific data. These events can be the result of human-computer interaction, such as a user launching an application, or they can be the result of the autonomous operation of the IT system (e.g., scheduled backup). Explicit traces directly record the occurrence of certain types of events as part of the normal operation of the system; most prominently, these include a variety of timestamped system and application event logs. Implicit traces take many forms, and allow the occurrence of some events to be deduced from the observed state of the system, or artifact, and engineering knowledge of how the system operates. For example, the presence on a storage device of a unique chunk of data that is part of a known file can demonstrate that the file was likely to have been present once, and was subsequently deleted and partially overwritten. The observed absence of normal log files can point to a security breach during which the perpetrators wiped the system logs as a means to cover their tracks.

Although they frequently exist, these traces of cyber interactions are the result of conscious engineering decisions that are *not* usually taken to specifically facilitate forensics. This has important implications with respect to the provenance and authenticity of digital evidence, given the ease with which digital information can be modified.

1.1 Legal Concerns and the Daubert Standard

The first published accounts of misuse and manipulation of computer systems for illegal purposes such as theft, espionage and other crimes date back to the 1960s. During the 1970s, the first empirical studies of computer crime were carried out using established criminological research methods. In the early-to-mid 1980s, targeted computer crime legislation emerged across Europe and North America [7, 8]; in recognition of the inherent cross-jurisdictional scope of many such crimes, international cooperation agreements were also put in place.

In the UK, the Computer Misuse Act 1990 [3] defines computer-specific crimes – S1 Unauthorised Access To Computer Material, S2 Unauthorised Access with Intent to Commit Other Offences, S3 Unauthorised Acts with Intent to Impair Operation, and S3A Making, Supplying or Obtaining. The Police & Criminal Evidence Act 1984 and Criminal Justice & Police Act 2001 address computer-specific concerns with respect to warrants, search and seizure.

In many jurisdictions, legal statutes related to misuse of telecommunications are separate (and older than) those related to computer crimes. We use the umbrella term *cybercrime* to collectively refer to all crimes related to computer and telecommunications misuse; broadly, these include the use of cyber systems to commit any type of crime, as well as the criminal targeting of cyber systems.

As is usually the case, legal systems require time to assimilate new laws and integrate them into routine law practice. Conversely, legislation usually requires corrections, clarification and unified interpretation in response to concerns encountered in the courtroom. One of the earliest and most influential legal precedents was set by the US supreme court, which used three specific cases – Daubert v. Merrell Dow Pharmaceuticals, 509 U.S. 579 (1993); General Electric Co. v. Joiner, 522 U.S. 136 (1997); and Kumho Tire Co. v. Carmichael, 526 U.S. 137 (1999) – to establish a new standard for the presentation of scientific evidence in legal proceedings, often referred to as the Daubert standard [9].

As per Goodstein [4], "The presentation of scientific evidence in a court of law is a kind of shotgun marriage between the two disciplines. ... The Daubert decision is an attempt (not the first, of course) to regulate that encounter." These cases set a new standard for expert testimony, overhauling the previous Frye standard of 1923 (Frye v. United States, 293 F. 1013, D.C. Cir. 1923). In brief, the supreme court instructed trial judges to become gatekeepers of expert testimony, and gave four basic criteria to evaluate the admissibility of forensic evidence:

1. The theoretical underpinnings of the methods must yield testable predictions by means of which the theory could be falsified.
2. The methods should preferably be published in a peer-reviewed journal.
3. There should be a known rate of error that can be used in evaluating the results.
4. The methods should be generally accepted within the relevant scientific community.

The court also emphasised that these standards are flexible and that the trial judge has a lot of leeway in determining the admissibility of forensic evidence and expert witness testimony. The Daubert criteria have been broadly accepted, in principle, by other jurisdictions subject to interpretation in the context of local legislation. In the UK, the Law Commission for England and Wales proposed in consultation paper No. 190 [5] the adoption of criteria that build on Daubert.

The *ACPO Good Practice Guide for Digital Evidence* codifies four basic principles for the acquisition and handling of digital evidence:

1. No action taken by law enforcement agencies, persons employed within those agencies or their agents should change data which may subsequently be relied upon in court.
2. In circumstances where a person finds it necessary to access original data, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions.
3. An audit trail or other record of all processes applied to digital evidence should be created and preserved. An independent third party should be able to examine those processes and achieve the same result.
4. The person in charge of the investigation has overall responsibility for ensuring that the law and these principles are adhered to.

These principles seek to provide operational guidance to digital forensic investigators on how to maintain the integrity of the evidence and the investigative process, such that the evidence can be used in a court of law.

In the UK, the *forensic science regulator* mandates that any provider of digital forensic science must be “accredited to BS EN ISO/IEC 17020:2012 for any crime scene activity and BS EN ISO/IEC 17025:2005 for any laboratory function (such as the recovery or imaging of electronic data)” [10]. ISO/IEC 17025 [11] is an international standard specifying general requirements for the competence of testing and calibration laboratories; in other words, the certification attests to the quality and rigour of the processes followed in performing the forensic examination.

In the US, there is no strict legal requirement for digital forensic science providers to be certified to particular standards. Most large federal and state forensic labs do maintain ISO 17025 certifications; as of 2019, eighty five of them have such credentials for the processing of digital evidence.

Digital forensic techniques are also applied in a much broader range of inquiries, such as internal corporate investigations, that often do not result in formal proceedings in public court. Despite the fact that investigations may not require the same standard of proof, forensic analysts should always follow sound forensic practices in collecting and analysing the artifacts. This includes adherence to any judicial requirements when working with inherently personal data, which can be a non-trivial concern when the investigation is multi-jurisdictional. In such cases, it is important to seek timely legal advice to preserve the integrity of the inquiry.

1.2 Definitions

In 2001, the first Digital Forensics Research Workshop (DFRWS) was organised in response to the need to replace the prevalent *ad hoc* approach to digital evidence with a systematic, multi-disciplinary effort to firmly establish digital forensics as a rigorous scientific discipline. The workshop produced an in-depth report outlining a research agenda and provided one of the most frequently cited definitions of digital forensic science in the literature:

> [DFRWS] **Digital forensics** *is 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 unauthorised actions shown to be disruptive to planned operations.* [12]

This definition, although primarily stressing the investigation of criminal actions, also includes an anticipatory element, which is typical of the notion of forensics in operational environments, and brings it closer to incident response and cyber defence activities. In these situations, the analysis is primarily to identify the vector of attack and the scope of a security incident; the identification of adversaries with any level of certainty is rare and prosecution is not the typical outcome. In contrast, the reference definition provided by NIST a few years later [6] is focused entirely on the legal aspects of forensics, and emphasises the importance of a strict chain of custody:

> [NIST] **Digital forensics** (NIST) *is considered the application of science to the identification, collection, examination, and analysis of data while preserving the integrity of the information and maintaining a strict chain of custody for the data. Data refers to distinct pieces of digital information that have been formatted in a specific way.* [6]

The above law-centric definitions provide a litmus test for determining whether specific investigative tools and techniques qualify as being forensic. From a legal perspective, a flexible, open-ended definition is normal and necessary during legal proceedings to fit the case. However, from a technical perspective, they do not provide a meaningful starting point; therefore, we can adapt a refinement of the working definition first introduced in [13]:

> [Working] **Digital forensics** *is the process of identifying and reconstructing the relevant sequence of events that have led to the currently observable state of a target IT system or (digital) artifacts.*

The notion of *relevance* is inherently case-specific, and a large part of forensic analysts' expertise is the ability to identify evidence that concerns the case at hand. Frequently, a critical component of forensic analysis is the causal attribution of event sequence to specific human actors of the system (such as users, administrators, attackers). The provenance, reliability, and integrity of the data used as evidence data is of *primary* importance.

According to this definition, we can view every effort made to perform system or artifact analysis after the fact as a form of digital forensics. This includes common activities such as incident response and internal investigations, which almost never result in any legal action. On balance, only a tiny fraction of forensic analyses make it to the courtroom as formal evidence, although this should not constrain us from exploring the full spectrum of techniques for reconstructing the past of digital artifacts. The benefit of employing a broader view of forensic computing is that it helps us to identify closely related tools and methods that can be adapted and incorporated into forensics.

1.3 Conceptual Models

In general, there are two possible approaches to rebuilding the relevant sequence of events in the analysis of a cyber system from the available data sources – *state*-centric, and *history*-centric/*log*-centric. The starting point for state-centric approaches is a snapshot of the state of the system of interest; for example, the current content of a hard drive or another storage medium. Using the knowledge of how a particular system/application operates, we can deduce a prior state of interest. For example, if unique pieces of a known file are on the medium, but the file is not available via the normal file system interface, the most likely explanation is that the file was once stored in the file system but was subsequently deleted (the space was marked for reuse) and partially overwritten by newer files. The main constraint here is the dearth of historical data points, which limits our ability to deduce the state of the system at any given point in the past.

Log-centric approaches rely on an explicit, timestamped history of events (a log) that documents the updates to the system's state. For example, a packet capture contains the complete history of network communications over a period of time. Operating Systems (OSs) maintain a variety of monitoring logs that detail various aspects of the operation of the OS kernel and different applications; additional auditing and security monitoring tools can provide yet more potentially relevant events. Many applications, especially in the enterprise domain, provide application-level logs. Thus, a log-rich environment contains potentially all the relevant details to an investigation; the challenge is to sift through the log entries, which often number in the millions, to find and put together the relevant events.

Historically, storage has been a precious resource in computer systems, leading to software designs that emphasise space efficiency by updating the information in place, and keeping a minimal amount of log information. Consequently, the principal approach to forensics has been predominantly state-centric.

Over the last ten to fifteen years, technology advances have made storage and bandwidth plentiful and affordable, which has led to a massive increase in the amount of log data maintained by IT systems and applications. There is a clear trend towards increasing the amount and granularity of telemetry data being sent over the network by operating systems and individual applications as part of their normal operations. Consequently, there is a substantial need to evolve a forensic methodology such that log information takes on a correspondingly higher level of importance. In other words, the current period marks an important evolution in digital forensic methodology, one that requires substantial retooling and methodological updates.

1.3.1 Cognitive Task Model

Differential analysis [14] is a basic building block of the investigative process, one that is applied at varying levels of abstraction and to a wide variety of artifacts. However, it does not provide an overall view of how forensic experts actually perform an investigation. This is particularly important in order to build forensic tools that better support cognitive processes.

Unfortunately, digital forensics has not been the subject of any serious interest on the part of cognitive scientists and there has been no coherent effort to document forensic investigations. Therefore, we adopt the sense-making process originally developed by Pirolli & Card [15] to describe intelligence analysis - a cognitive task that is very similar to forensic analysis. The Pirolli & Card cognitive model is derived from an in-depth Cognitive Task Analysis (CTA), and provides a reasonably detailed view of the different aspects of an intelligence analyst's work. Although many of the tools are different, forensic and intelligence analysis are very similar in nature - in both cases analysts have to go through a mountain of raw data to identify (relatively few) relevant facts and put them together into a coherent story. The benefit of using this model is that: a) it provides a fairly accurate description of the investigative process in its own right, and allows us to map the various tools to the different phases of the investigation; b) it provides a suitable framework for explaining the relationships of the various models developed within the area of digital forensics; and c) it can seamlessly incorporate information from other lines of the investigation.

The overall process is shown in Figure 1. The rectangular boxes represent different stages in the information processing pipeline, starting with raw data and ending with presentable results. The arrows indicate transformational processes that move information from one box to another. The x axis approximates the overall level of effort required to move information from the raw to the specific processing stage. The y axis shows the amount of structure (with respect to the investigative process) in the processed information for every stage. Thus, the overall trend is to move the relevant information from the lower left-hand to the upper right-hand corner of the diagram. In reality, the processing can both meander through multiple iterations of local loops and jump over phases (for routine cases handled by an experienced investigator).

External data sources include all potential evidence sources for a specific investigation such as disk images, memory snapshots, network captures and reference databases such as hashes of known files. The *shoebox* is a subset of all the data that have been identified as potentially relevant, such as all the email communications between two persons of interest. At any given time, the contents of the shoebox can be viewed as the analyst's approximation of the information content that is potentially relevant to the case. The *evidence file* contains only the parts that are directly relevant to the case such as specific email exchanges on a topic of interest.

The *schema* contains a more organised version of the evidence such as a timeline of events or a graph of relationships, which allows higher-level reasoning over the evidence. A *hypothesis* is a tentative conclusion that explains the observed evidence in the schema and, by extension, could form the final conclusion. Once the analyst is satisfied that the hypothesis is supported by the evidence, the hypothesis turns into a *presentation*, which is the final product of the process. The presentation usually takes on the form of an investigator's report that both speaks to the high-level conclusions that are relevant to the legal case and also documents the low-level technical steps based on which the conclusion has been formed.

The overall analytical process is *iterative* in nature with two main activities loops: a *foraging*

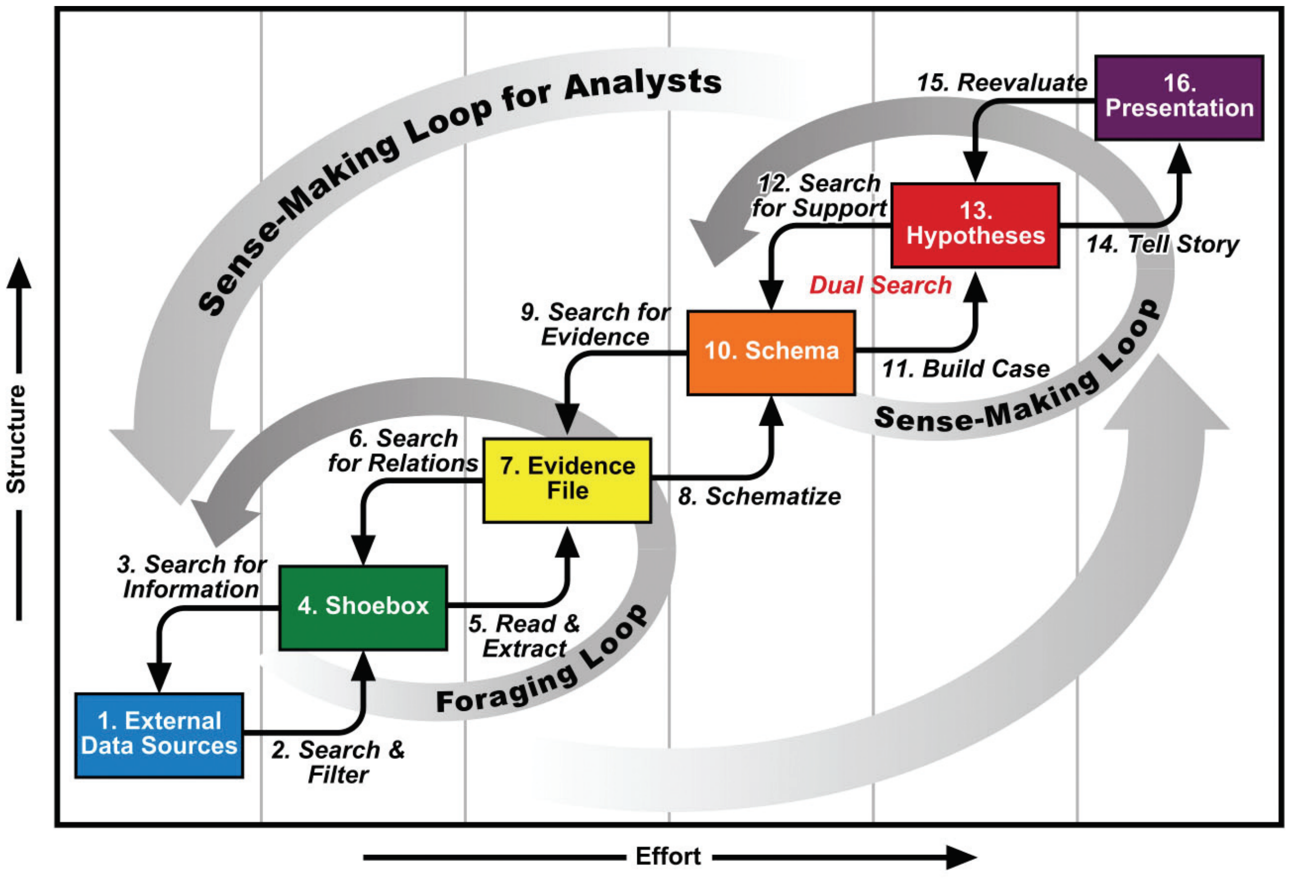

Figure 1: Notional model of sense-making loop for analysts derived from cognitive task analysis [16, p.44].

loop that involves the actions taken to find potential sources of information, and which then queries them and filters them for relevance; and a *sense-making loop* in which the analyst develops – in an iterative fashion – a conceptual model that is supported by the evidence. The information transformation processes in the two loops can be classified into bottom-up (organising data to build a theory) or top-down (finding data based on a theory) ones. Analysts apply these techniques in an opportunistic fashion with many iterations, in response to both newly discovered pieces of evidence, and to high-level investigative questions.

1.3.2 Bottom-Up Processes

Bottom-up processes are synthetic – they build higher-level (more abstract) representations of information from more specific pieces of evidence.

- *Search and filter*: External data sources, hard drives, network traffic, etc. are searched for relevant data based on keywords, time constraints and others in an effort to eliminate the vast majority of irrelevant data.
- *Read and extract*: Collections in the shoebox are analysed to extract individual facts and relationships that can support or disprove a theory. The resulting pieces of artifacts (e.g., individual email messages) are usually annotated with their relevance to the case.
- *Schematize*: At this step, individual facts and simple implications are organised into a schema that can help organise and identify the significance of and relationships between a growing number of facts and events. Timeline analysis is one of the basic tools of

the trade; however, any method of organising and visualising the facts-graphs, charts, etc.-can greatly speed up the analysis. This is not an easy process to formalise, and most forensic tools do not directly support it. Therefore, the resulting schemas may exist on a piece of paper, on a whiteboard or only in the mind of the investigator. Since the overall case could be quite complicated, individual schemas may cover specific aspects of it such as the discovered sequence of events.

- *Build case*: From the analysis of the schemas, the analyst eventually comes up with testable theories, or *working hypotheses*, that can explain the evidence. A working hypothesis is a tentative conclusion and requires more supporting evidence, as well as rigorous testing against alternative explanations. It is a central component of the investigative process and is a common point of reference that brings together the legal and technical sides in order to build a case.
- *Tell story*: The typical result of a forensic investigation is a final report and, perhaps, an oral presentation in court. The actual presentation may only contain the part of the story that is strongly supported by the digital evidence; weaker points may be established by drawing on evidence from other sources.

1.3.3 Top-Down Processes

Top-down processes are analytical – they provide context and direction for the analysis of less structured data search and they help organise the evidence. Partial or tentative conclusions are used to drive the search for supporting and contradictory pieces of evidence.

- *Re-evaluate*: Feedback from clients may necessitate re-evaluations, such as collecting stronger evidence or pursuing alternative theories.
- *Search for support*: A hypothesis may need more facts to be of interest and, ideally, would be tested against every (reasonably) possible alternative explanation.
- *Search for evidence*: Analysis of theories may require the re-evaluation of evidence to ascertain its significance/provenance, or it may trigger the search for more/better evidence.
- *Search for relations*: Pieces of evidence in the file can suggest new searches for facts and relations on the data.
- *Search for information*: The feedback loop from any of the higher levels can ultimately cascade into a search for additional information; this may include new sources, or the re-examination of information that was filtered out during previous passes.

1.3.4 The Foraging Loop

It has been observed [17] that analysts tend to start with a high-recall/low-selectivity query, which encompasses a fairly large set of documents – many more than the analyst can read. The original set is then successively modified and narrowed down before the documents are read and analysed.

The *foraging loop* is a balancing act between three kinds of processing that an analyst can perform- *explore*, *enrich* and *exploit*. Exploration effectively expands the shoebox by including larger amounts of data; enrichment shrinks it by providing more specific queries that include fewer objects for consideration; exploitation is the careful reading and analysis of an artifact to extract facts and inferences. Each of these options has varying costs and potential rewards and, according to information foraging theory [18], analysts seek to optimise their cost/benefit trade-offs.

Information foraging in this context is a highly iterative process with a large number of incremental adjustments in response to the emerging evidence. It is the responsibility of the investigator to keep the process on target and within the boundaries of any legal restrictions.

1.3.5 The Sense-Making Loop

Sense-making is a cognitive term and, according to Klein's [19] widely quoted definition, is the ability to make sense of an ambiguous situation. It is the process of creating situational awareness and understanding to support decision making in the face of uncertainty – an effort to understand connections between people, places and events in order to anticipate their trajectories and act effectively.

There are three main processes involved in the sense-making loop: *problem structuring*-the creation and exploration of hypotheses, *evidentiary reasoning* – the employment of evidence to support/disprove a hypothesis and *decision making*-selecting a course of action from a set of available alternatives.

It is important to recognize that the described information processing loops are closely tied together and often trigger iterations in either directions. New evidence may require a new working theory, whereas a new hypothesis may drive the search for new evidence to support or disprove it.

1.3.6 Data Extraction vs. Analysis vs. Legal Interpretation

Considering the overall process from Figure 1, we gain a better understanding of the roles and relationships among the different actors. At present, digital forensic researchers and tool developers primarily provide the means to acquire the digital evidence from the forensic targets, extract (and logically reconstruct) data objects from it, and the essential tools to search, filter, and organize it. In complex cases, such as a multi-national security incident, identifying and acquiring the relevant forensic targets can be a difficult and lengthy process. It is often predicated in securing the necessary legal rulings in multiple jurisdictions, as well as the cooperation of multiple organizations.

Forensic investigators are the primary users of these technical capabilities, employing them to analyse specific cases and to present legally-relevant conclusions. It is the responsibility of the investigator to drive the process and to perform all the information foraging and sense-making tasks. As the volume of the data being analysed continues to grow, it becomes ever more critical for the forensic software to offer higher levels of automation and abstraction. Data analytics and natural language processing methods are starting to appear in dedicated forensic software, and – going forward – an expanding range of statistical and machine learning tools will need to be incorporated into the process.

Legal experts operate in the upper right-hand corner of the depicted process in terms of building/disproving legal theories. Thus, the investigator's task can be described as the translation of highly specific technical facts into a higher level representation and theory that explains them. The explanation is almost always connected to the sequence of the actions of the people that are part of the case, such as suspects, victims, and witnesses.

In summary, investigators need not be forensic software engineers, but they must be technically proficient enough to understand the significance of the artifacts extracted from data sources, and they must be able to read the relevant technical literature (peer-reviewed articles) in full. As the sophistication of the tools grows, investigators will need to have a working understanding of a growing list of data science methods that are employed by the tools in order to correctly interpret the results. Similarly, analysts must have a working understanding of the legal landscape, and they must be able to produce a competent report and present their findings on the witness stand, if necessary.

1.3.7 Forensic Process

The defining characteristic of forensic investigations is that their results must be admissible in court. This entails following established procedures for acquiring, storing, and processing of the evidence, employing scientifically established analytical tools and methods, and strict adherence to a professional code of practice and conduct.

Data Provenance and Integrity. Starting with the data acquisition process, an investigator must follow accepted standards and procedures in order to certify the provenance and maintain the integrity of the collected evidence. In brief, this entails acquiring a truthful copy of the evidence from the original source using validated tools, keeping custodial records and detailed case notes, using validated tools to perform the analysis of the evidence, cross-validating critical pieces of evidence, and correctly interpreting the results based on peer-reviewed scientific studies.

As discussed in the following section, data acquisition can be performed at different levels of abstraction and completeness. The traditional gold standard is a bit-level copy of the forensic target, which can then be analysed using knowledge of the structure and semantics of the data content. As storage devices increase in complexity and encryption becomes the default data encoding, it is increasingly infeasible to obtain a true physical copy of the media and a (partial) logical acquisition may be the only possibility. For example, the only readily available source of data content for an up-to-date smartphone (with encrypted local storage) might be a cloud backup of the user's data. Further, local data may be treated by the courts as having higher levels of privacy protection than data shared with a third party, such as a service provider.

Scientific Methodology. The notion of *reproducibility* is central to the scientific validity of forensic analysis; starting with the same data and following the same process described in the case notes should allow a third party to arrive at the same result. Processing methods should have scientifically established error rates and different forensic tools that implement the same type of data processing should yield results that are either identical, or within known statistical error boundaries.

The investigator must have a deep understanding of the results produced by various forensic computations. Some of the central concerns include: inherent uncertainties in some of the source data, the possibility for multiple interpretations, as well as the recognition that some of the data could be fake in that it was generated using anti-forensics tools in order to confuse the investigation. The latter is possible because most of the data item used in the forensic analysis is produced during the normal operation of the system, and is not tamper-proof. For example, an intruder with sufficient access privileges can arbitrarily modify any of the millions of file timestamps potentially making timeline analysis – a core analytical technique – unreliable. Experienced forensic analysts are alert to such issues and seek, whenever possible, to corroborate important pieces of information from multiple sources.

Tool Validation. Forensic tool validation is a scientific and engineering process that subjects specific tools to systematic testing in order to establish the validity of the results produced. For example, data acquisition software must reliably produce an unmodified and complete copy of the class of forensic targets it is designed to handle.

Forensic Procedure. The organizational aspect of the forensic process, which dictates how evidence is acquired, stored, and processed is critical to the issue of admissibility. Strict adherence to established standards and court-imposed restriction is the most effective means of demonstrating to the court that the results of the forensic analysis are truthful and trustworthy.

Triage. The volume of data contained by a forensic target typically far exceeds the amount of data *relevant* to an inquiry. Therefore, in the early stages of an investigation, the focus of the analysis is to (quickly) identify the relevant data and filter out the irrelevant. Such initial screening of the content, often referred to as *triage*, results in either follow up deep examination, or in deprioritisation, or removal of the target from further consideration.

Legally, there can be a number of constraints placed on the triage process based on the the case and the inherent privacy rights in the jurisdiction. From a technical perspective [20], "triage is a partial forensic examination conducted under (significant) time and resource constraints." In other words, investigators employ fast examination methods, such as looking at file names, examining web search history, and similar, to estimate (based on experience) the value of the data. Such results are inherently less reliable than a deep examination as it is easy to create a mismatch between data attribute and actual content. Therefore, courts may place constraints on the use of computers by convicted offenders to facilitate fast screening by officers in the field without impounding the device.

2 OPERATING SYSTEM ANALYSIS

[6, 21, 22]

Modern computer systems generally still follow the original von Neumann architecture [23], which models a computer system as consisting of three main functional units – CPU, main memory, and secondary storage – connected via data buses. To be precise, the *actual* investigative targets are not individual pieces of hardware, but the different Operating System (OS) modules controlling the hardware subsystems and their respective data structures.

Our discussion takes a high level view of OS analysis – it is beyond the scope of the Knowledge Area to delve into the engineering details of how different classes of devices are analysed. For example, smartphones present additional challenges with respect to data acquisition;

however, they are still commodity computers with the vast majority of them running on a *Linux* kernel. The same applies to other classes of embedded devices, such as UAVs and vehicle infotainment systems.

The OS functions at a higher level of privilege relative to user applications and directly manages all the computer system's resources – CPU, main memory, and I/O devices. Applications request resources and services from the OS via the system call interface and employ them to utilize them to accomplish a specific task. The (operating) system maintains a variety of accounting information that can bear witness to events relevant to an inquiry [21].

System analysis employs knowledge of how operating systems function in order to reach conclusions about events and actions of interest to the case. Average users have very little understanding of the type of information operating systems maintain about their activities, and usually do not have the knowledge and/or privilege level to tamper with system records thereby making them forensically useful, even if they do not fit a formal definition for secure and trustworthy records.

2.1 Storage Forensics

Persistent storage in the form of Hard Disk Drives (HDDs), Solid State Drives (SSDs), optical disks, external (USB-connected) media etc. is the primary source of evidence for most digital forensic investigations. Although the importance of (volatile) memory forensics in solving cases has grown significantly, a thorough examination of persistent data has remained a cornerstone of most digital forensic investigations.

2.1.1 Data Abstraction Layers

Computer systems organise raw storage in successive layers of abstraction – each software layer (some may be in firmware) builds an incrementally more abstract data representation that is only dependent on the interface provided by the layer immediately below it. Accordingly, forensic analysis of storage devices can be performed at several levels of abstraction [22]:

PHYSICAL MEDIA. At the lowest level, every storage device encodes a sequence of bits and it is possible, in principle, to use a custom mechanism to extract the data bit by bit. Depending on the underlying technology, this can be an expensive and time-consuming process, and often requires reverse engineering. One example of this process is the acquisition of mobile phone data, in some of which it is possible to physically remove (desolder) the memory chips and perform a true hardware-level acquisition of the content [24]. A similar "chip-off" approach can be applied to a flash memory devices, such as SSD, and to embedded and Internet of Things (IoT) devices with limited capabilities and interfaces. Another practical approach is to employ engineering tools that support the hardware development process and employ, for example, a standard JTAG interface [25] – designed for testing and debugging purposes – to perform the necessary data acquisition.

In practice, the lowest level at which typical examinations are performed is the Host Bus Adapter (HBA) interface. Adapters implement a standard protocol (SATA, SCSI) through which they can be made to perform low-level operations, such as accessing the drive's content. Similarly, the NVMe protocol [26] is used to perform acquisition from PCI Express-based solid-state storage devices.

All physical media eventually fail and (part of) the stored data may become unavailable.

Depending on the nature of the failure, and the sophistication of the device, it may be possible to recover at least some of the data. For example, it may be possible to replace the failed controller of a HDD and recover the content. Such hardware recovery becomes more difficult with more integrated and complex devices.

BLOCK DEVICE. The typical HBA presents a block device abstraction – the medium is presented as a sequence of fixed-size blocks, commonly consisting of 512 or 4096 bytes, and the contents of each block can be read or written using block read/write commands. The typical data acquisition process works at the block device level to obtain a working copy of the forensic target – a process known as *imaging* – on which all further processing is performed. Historically, the term *sector* is used to refer to the data transfer units of magnetic hard disks; a (logical) block is a more general term that is independent of the storage technology and physical data layout.

FILE SYSTEM. The block device has no notion of files, directories or – in most cases – which blocks are considered allocated and which ones are free; it is the filesystem's task to organise the block storage into a file-based store in which applications can create files and directories with all of their relevant metadata attributes – name, size, owner, timestamps, access permissions etc.

APPLICATION ARTIFACTS. User applications use the filesystem to store various artifacts that are of value to the end-user – documents, images, messages etc. The operating system itself also uses the file system to store its own image – executable binaries, libraries, configuration and log files, registry entries – and to install applications. Some application artifacts such as compound documents have a complex internal structure integrating multiple artifacts of different types. An analysis of application artifacts tends to yield the most immediately relevant results, as the recorded information most directly relates to actions and communications initiated by people. As the analysis goes deeper (to a lower level of abstraction), it requires greater effort and more expert knowledge to independently reconstruct the actions of the system. For example, by understanding the on-disk structures of a specific filesystem, a tool can reconstitute a file out of its constituent blocks. This knowledge is particularly costly to obtain from a closed system such as Microsoft Windows, because of the substantial amount of blackbox reverse engineering involved.

Despite the cost, independent forensic reconstruction is of critical importance for several reasons:

- It enables the recovery of evidentiary data not available through the normal data access interface.
- It forms the basis for recovering partially overwritten data.
- It allows the discovery and analysis of malware agents that have subverted the normal functioning of the system, thus making the data obtained via the regular interface untrustworthy.

2.2 Data Acquisition

In line with best practices [6], analysing data at rest is not carried out on a live system. The target machine is powered down, an exact bit-wise copy of the storage media is created, the original is stored in an evidence locker and all the forensic work is performed on the copy. There are exceptions to this workflow in cases where it is not practical to shut down the target system and, therefore, a media image is obtained while the system is live. Evidently, such an approach does not provide the same level of consistency guarantees, but it can still yield valuable insights. The problem of consistency, also referred to as *data smearing*, does not exist in virtualised environments, where a consistent image of the virtual disk can be trivially obtained by using the built-in snapshot mechanism.

As already discussed, obtaining data from the lowest level system interface available and independently reconstructing higher-level artifacts is considered the most reliable approach to forensic analysis. This results in a strong preference for acquiring data at lower levels of abstraction and the concepts of *physical* and *logical* acquisition.

Physical data acquisition is the process of obtaining the data directly from hardware media, without the mediation of any (untrusted) third-party software.

An increasingly common example of this approach is mobile phone data acquisition that relies on removing the physical memory chip[24] and reading the data directly from it. More generally, getting physical with the evidence source is usually the most practical and necessary method for low-end embedded systems with limited hardware capabilities. Physical acquisition also affords access to additional over-provisioned raw storage set aside by the storage device in order to compensate for the expected hardware failures. As a general rule, devices offer no external means to interrogate this shadow storage area.

Chip-off techniques present their own challenges in that the process is inherently destructive to the device, the data extraction and reconstruction requires additional effort, and the overall cost can be substantial.

For general-purpose systems, tools use an HBA protocol such as SATA or SCSI to interrogate the storage device and obtain a copy of the data. The resulting image is a block-level copy of the target that is generally referred to as physical acquisition by most investigators; Casey uses the more accurate term *pseudo*-physical to account for the fact that not every area of the physical media is acquired and that the order of the acquired blocks does not necessarily reflect the physical layout of the device.

In some cases, it is necessary to perform additional recovery operations before a usable copy of the data is obtained. One common example is RAID storage devices, which contain multiple physical devices that function *together* as a single unit, providing built-in protection against certain classes of failures. In common configurations such as RAID 5 and 6 the content acquisition of individual drives is largely useless without the subsequent step of RAID data reconstruction.

Modern storage controllers are quickly evolving into autonomous storage devices, which implement complex (proprietary) wear-levelling and load-balancing algorithms. This has two major implications: a) the numbering of the data blocks is completely separated from the actual physical location; and b) it is possible for the storage controller itself to become compromised [27], thus rendering the acquisition process untrustworthy. These caveats notwithstanding, we will refer to block-level acquisition as being physical, in line with the accepted terminology.

Logical data acquisition relies on one or more software layers as intermediaries to acquire the data from the storage device.

In other words, the tool uses an Application Programming Interface (API), or message protocol, to perform the task. The integrity of this method hinges on the correctness and integrity of the implementation of the API, or protocol. In addition to the risk, however, there is also a reward – higher level interfaces present a data view that is closer in abstraction to that of user actions and application data structures. Experienced investigators, if equipped with the proper tools, can make use of both physical and logical views to obtain and verify the evidence relevant to the case.

Block-level acquisition can be accomplished in software, hardware or a combination of both. The workhorse of forensic imaging is the dd Unix/Linux general purpose command-line utility, which can produce a binary copy of any file, device partition or entire storage device. A hardware write blocker is often installed on the target device to eliminate the possibility of operator error, which can lead to the accidental modification of the target.

Cryptographic hashes are computed for the entire image and (preferably) for every block; the latter can be used to demonstrate the integrity of the remaining evidence if the original device suffers a partial failure, which makes it impossible to read its entire contents. The National Institute of Standards and Technology (NIST) maintains the Computer Forensic Tool Testing (CFTT) project [28], which independently tests various basic tools, such as write blockers and image acquisition tools and regularly publishes reports on its findings.

Encryption Concerns

Apart from having the technical capability to safely interrogate and acquire the content of a storage device, one of the biggest concerns during data acquisition can be the presence of encrypted data. Modern encryption is pervasive and is increasingly applied by default to both stored data and data in transit over the network. By definition, a properly implemented and administered data security system, which inevitably employs encryption, will frustrate efforts to acquire the protected data and, by extension, to perform forensic analysis.

There are two possible paths to obtaining encrypted data – technical and legal. The technical approach relies on finding algorithmic, implementation, or administrative errors, which allow the data protection to be subverted. Although it is nearly impossible to create a complex IT system that has no bugs, the discovery and exploitation of such deficiencies is becoming increasingly more difficult and resource intensive.

The legal approach relies on compelling the person with knowledge of the relevant encryption keys to surrender them. This is relatively new legal territory and its treatment varies across jurisdictions. In the UK, the *Regulation of Investigatory Powers Act 2000* specifies the circumstances under which individuals are legally required to disclose the keys. Disclosure may run counter the legal right against self-incrimination and in some jurisdictions, such as in the United States, it is not yet definitively resolved.

The remainder of this discussion assumes that access to the raw data is ensured by either technical, or legal means, which are beyond the scope of this knowledge area.

2.3 Filesystem Analysis

A typical storage device presents a block device interface with B_{max} number of blocks of size B_{size}. All read and write I/O operations are executed at the granularity of a whole block; historically, the standard block size adopted by HDD manufacturers has been 512 bytes. With the 2011 introduction of the advanced format standard [29], storage devices can support larger blocks, with 4,096 bytes being the new preferred size.

Regardless of the base block size, many operating systems manage storage in clusters; a *cluster* is a contiguous sequence of blocks and is the smallest unit at which raw storage is allocated/reclaimed. Thus, if the device block/sector size is 4KiB but the chosen cluster size is 16KiB, the OS will allocate blocks in groups of four.

For administration purposes, the raw drive may be split into one or more contiguous areas called *partitions*, each of which has a designated use and can be independently manipulated. Partitions can further be organised into volumes – a *physical volume* maps onto a single partition, whereas a *logical volume* can integrate multiple partitions potentially from multiple devices. Volumes present a block device interface but allow for the decoupling of the physical media organisation from the logical view presented to the operating system.

With a few exceptions, volumes/partitions are formatted to accommodate a particular file system (filesystem), which organizes and manages the blocks to create the abstraction of files and directories together with their relevant metadata. The Operating System (OS), as part of its system call interface used by applications to request services, provides a filesystem API that allows applications to create, modify and delete files; it also allows files to be grouped into a hierarchical structure of directories (or folders).

A **file** is a named (opaque) sequence of bytes that is stored persistently.

As a general rule, the format and interpretation of file content is almost always outside the purview of the operating system; it is the concern of relevant applications acting on behalf of users.

A **file system** (filesystem) is an OS subsystem that is responsible for the persistent storage and organisation of user and system files on a partition/volume.

It provides a high-level standard API such as POSIX, that is used by applications to store and retrieve files by name without any concern for the physical storage method employed or the layout of the data (and metadata) content.

Filesystem forensics uses knowledge of the filesystem's data structures and the algorithms used to create, maintain, and delete them to: a) extract data content from devices independently of the operating system instance which created it; and b) extract leftover artifacts to which the regular filesystem API does not offer access.

The first feature is important to ensure that the data are not being modified during acquisition and that any potential security compromises do not affect the validity of the data. The second provides access to (parts of) deallocated files that have not been overwritten, purposely hidden data, and an implied history of the filesystem operation – the creation/deletion of files – that is not explicitly maintained by the OS.

2.4 Block Device Analysis

Before the OS can organise a filesystem on a raw device, it typically splits it into a set of one or more disjoint *partitions*.

> A block device **partition**, or *physical volume*, is a contiguous allocation of blocks for a specific purpose, such as the organisation of a file system.

Partitions are the basic method used for coarse-grained storage management; they allow a single physical device to be dedicated to multiple purposes such as hosting different filesystems or separating system from user files. If a subdivision is not needed, the entire device can be trivially allocated to a single partition.

> A **logical volume** is a collection of physical volumes presented and managed as a single unit.

Logical volumes allow storage capacity from different devices to be pooled transparently (to the filesystem) to simplify the use of available capacity. They also enable automated block-level replication in the form of RAIDs [30] for enhanced performance and durability.

2.5 Data Recovery & File Content Carving

One of the early staples of data recovery tools was the 'undelete' functionality, which can reverse the effects of users deleting data. The most common case is that of users deleting a file and needing to reverse the operation. On a HDD, this reversal is readily achievable immediately after the deletion - the storage taken up by the file's content is merely deallocated (marked as available), but no actual destruction (sanitisation) of the data takes place.

A more difficult case is a HDD that has been in use for some time and that has been subsequently formatted (e.g., by somebody attempting to destroy evidence). The often employed quick format command has the effect of overlaying a set of data structures that correspond to an empty filesystem (a full format sanitizes the content of the media but can take hours to complete so it is used less frequently). Thus, the normal filesystem interface, after querying these structures, will report that there are no files. The reality is that – at that moment – only filesystem metadata has been partially overwritten, and all the data blocks representing the file content are still present on the media in full.

Forensic computing, unlike most other types of computation, is very interested in all recoverable (partial) artifacts, including (and sometimes especially) deallocated ones. Unless a user has taken special measures to securely wipe a hard disk, at any given time the media contains recoverable application artifacts (files) that have ostensibly been deleted. The process of restoring the artifacts is commonly accomplished by *carving*.

> **File (content) carving** is the process of recovering and reconstructing file content directly from block storage without using the filesystem metadata. More generally, **data (structure) carving** is the process of reconstructing logical objects (such as files and database records) from a bulk data capture (disk/RAM image) without using metadata that describes the location and layout of the artifacts.

File carving is the oldest and most commonly used, technique and its basic form is based on

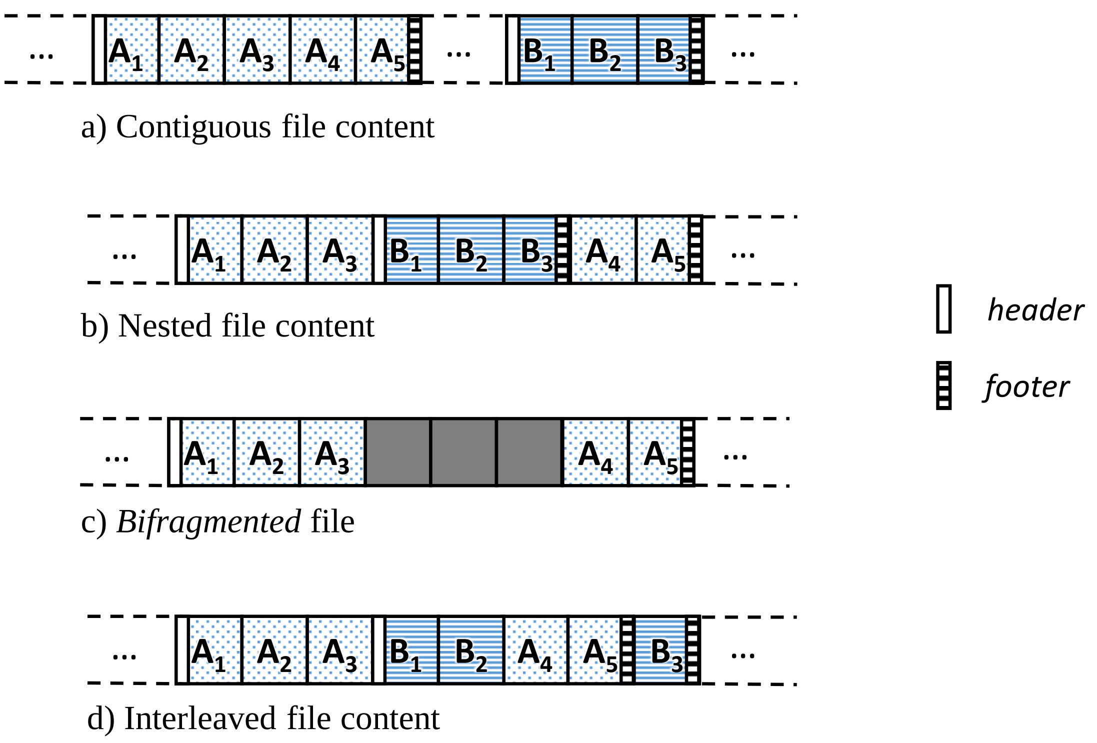

Figure 2: Common file content layout encountered during carving.

two simple observations: a) most file formats have specific beginning and end tags (a.k.a. *header* and *footer*); and b) file systems strongly favour a sequential file layout to maximise throughput.

Put together, these yield a basic recovery algorithm: 1) scan the capture sequentially until a known header is found; for example, JPEG images always start with the (hexadecimal) FF D8 FF header; 2) scan sequentially until a corresponding footer is found; FF D9 for JPEG; 3) copy the data in between as the recovered artifact. Figure 2 illustrates some of the most common cases encountered during file carving:

1. *No fragmentation* is the most typical case, as modern filesystems require extra effort to ensure sequential layout for optimal performance.
2. *Nested content* is often the result of deletion; in the example, after the initial sequential back-to-back layout of the files, the content ahead and behind file B was deleted and replaced by A. In some cases, the file format allows nesting; e.g., JPEGs commonly have a thumbnail version of the image, which is also in JPEG format. This case can be solved by making multiple passes – once B is carved out (and its blocks removed from further consideration) the content of A becomes contiguous, so a subsequent pass will readily extract it.
3. *Bi-fragmented files* are split into two contiguous pieces with the other content in between, which also determines how difficult the reconstruction is; if the content in the middle is easily distinguished from the content of the file (e.g., the pieces of text in the middle of a compressed image) then the problem is relatively easy. Otherwise, it is ambiguous and it could be quite difficult to identify the matching pieces.
4. *Interleaved content* is a more complicated version of nesting which happens when larger files are used to fill the gaps created by the deletion of smaller ones.

This simple carving approach usually yields a good number of usable artifacts; however, real data can contain a number of atypical patterns, which can lead to a large number of repetitive and/or false positive results. One major reason is that file formats are not designed with carving in mind and rarely have robust internal metadata that connect the constituent pieces together. Some do not even have a designated header and/or footer, and this can result in a large number of false positives, potentially producing results substantially larger in volume than the source data.

Slack space recovery. Both RAM and persistent storage are almost always allocated in multiples of a chosen minimum allocation units. Therefore, at the end of the allocated space, there is storage capacity – slack space – that is not used by the application, but is also *not* available for other uses. For example, if the minimum allocation is 4KiB, and a file needs 14KiB, the filesystem will allocate four 4KiB blocks. The application will fully use the first three blocks, but will only use 2KiB from the last block. This creates the potential to store data that would be inaccessible via the standard filesystem interface and can provide a simple means to hide data.

Slack space is the difference between the *allocated* storage for a data object, such as file, or a volume, and the storage in actual use.

Once aware of the potential for storing hidden data in slack space, it is relatively easy to identify and examine it, and this is a standard step in most investigations.

Upcoming challenges. As solid state drives continue to grow in capacity and displace hard disks from an increasing proportion of operational data storage, file carving's utility is set to diminish over time. The reason lies in the fact that SSD blocks need to be written twice in order to be reused (the first write resets the state of the block, thereby enabling its reuse). To improve performance, the TRIM and UNMAP commands were added to the ATA and SCSI command sets, respectively; they provide a mechanism for the filesystem to indicate to the storage device which blocks need to be garbage collected and prepared for reuse.

King & Vidas [31] established experimentally that file carving would only work in a narrow set of circumstances on modern Solid State Drives (SSDs). Specifically, they show that for a TRIM-aware operating system, such as Windows 7 and after, the data recovery rates in their tests were almost universally zero. In contrast, using a pre-TRIM OS (Windows XP) allows for near-perfect recovery rates under the same experimental conditions.

3 MAIN MEMORY FORENSICS

[32]

The early view of best forensic practices was to literally pull the plug on a machine that was to be impounded. The rationale was that this would remove any possibility of alerting the processes running on the host and would preempt any attempts to hide information. Over time, experience has shown that these concerns were largely exaggerated and that the substantial and irreversible loss of important forensic information such as open connections and encryption keys was rarely justified. Studies have clearly demonstrated that data tend to persist for a long time in volatile memory ([33], [34]). There is a wealth of information about a system's run-time state that can be readily extracted, even from a snapshot [32]:

Process information. It is practical to identify and enumerate all the running processes, threads and loaded systems modules; we can obtain a copy of the individual processes' code, stack, heap, code, and data segments. All this is particularly useful when analysing compromised machines, as it allows the identification of suspicious services, abnormal parent/child relationships, and, more generally, to search for known symptoms of compromise, or patterns of attack.

File information. It is practical for identifying any open files, shared libraries, shared memory, and anonymously mapped memory. This is particularly useful for identifying correlated user actions and file system activities, potentially demonstrating user intent.

Network connections. It is practical for identifying open and recently closed network connections and protocol information, as well as sending and receiving queues of data not yet sent or delivered, respectively. This information could readily be used to identify related parties and communication patterns between them.

Artifacts and fragments. Just like the filesystem, the memory management system tends to be reactive and leaves a lot of artifact traces behind. This is primarily an effort to avoid any processing that is not absolutely necessary for the functioning of the system; caching disk and network data tends to leave traces in memory for a long time.

Memory analysis can be performed either in real time on a live (running) system, or it could be performed on a snapshot (memory dump) of the state of the system. In addition to using specialized memory acquisitions tools, or a build-in snapshot mechanism (in virtualized environments) memory content can also be obtained from a system hibernation file, page swap, or a crash dump.

In live forensics, a trusted agent (process) designed to allow remote access over a secure channel is pre-installed on the system. The remote operator has full control over the monitored system and can take snapshots of specific processes, or the entire system. Live investigations are an extension of regular security preventive mechanisms, which allow for maximum control and data acquisition; they are primarily used in large enterprise deployments.

The main conceptual problem of working on a live system is that, if it is compromised, the data acquisition and analysis results are not trustworthy; therefore, forensic analysis is most frequently performed on a snapshot of the target system's RAM. Analysing a snapshot is considerably more difficult than working with a live system, which provides access to the state of the running system via a variety of APIs and data structures. In contrast, a raw memory capture offers no such facilities and forensic tools need to rebuild the ability to extract semantic information from the ground up. This is a semantic gap problem, and the purpose of memory forensics is to bridge it.

4 APPLICATION FORENSICS

Application forensics is the process of establishing a data-centric theory of operation for a specific application. The goal of the analysis is to objectively establish causal dependencies between data input and output, as a function of the user interactions with the application. Depending on whether an application is an open or closed source and on the level of the accompanying documentation, the analytical effort required can vary from reading detailed specifications to reverse engineering code, data structures and communication protocols, to performing time-consuming black box differential analysis experiments. Alternatively, forensic tool vendors may license code from the application vendor to gain access to the proprietary data structures.

The big advantage of analysing applications is that we have a better chance of observing and documenting direct evidence of user actions, which is of primary importance to the legal process. Also, the level of abstraction of the relevant forensic traces tend to have a level of abstraction corresponding to a particular domain.

4.1 Case Study: the Web Browser

Although there are at least four major web browsers in common use, after more than 20 years of development, their capabilities have converged, thus allowing us to talk about them in common terms. There are six main sources of forensically interesting information:

URL/search history. At present, there are no practical barriers to maintaining a complete browsing history (a log of visited websites), and making it available to users is a major usability feature; most users rarely delete this information. Separately, service providers such as Google and Facebook, are interested in this information for commercial reasons, and make it easy to share a browsing log with multiple devices. Combined with the content of the local file cache, the browsing history allows an investigator to almost look over the shoulder of the user of interest as they were navigating the Web. In particular, analysing user queries to search engines is among the most commonly employed techniques. The search query is encoded as part of the URL, and can often provide very clear and targeted clues as to what the user was trying to accomplish.

Form data. Browsers offer the convenience of remembering auto-completing passwords and other form data (such as address information). This can be very helpful to an investigator, especially if the user is less security conscious and does not use a master password to encrypt all of this information.

Temporary files. The local file cache provides its own chronology of web activities, including a stored version of the actual web objects that were downloaded and shown to the user (these may no longer be available online). Although caching has become considerably less effective owing to the increased use of dynamic content, this is tempered by the large increase in available storage capacity, which places very few, if any, practical constraints on the amount of data cached.

Downloaded files are, by default, never deleted providing another valuable source of activity information.

HTML5 local storage provides a standard means for web applications to store information locally; for example, this could be used to support disconnected operations, or to provide a measure of persistence for user input. Accordingly, the same interface can be interrogated to reconstruct web activities.

Cookies are opaque pieces of data used by servers to keep a variety of information on the web client in order to support transactions such as web mail sessions. In practice, most cookies are used by websites to track user behaviour, and it is well-documented that some providers go to great lengths to make sure that this information is resilient. Some cookies are time-limited access tokens that can provide access to online accounts (until they expire); others have a parsable structure and may provide additional information.

Most local information is stored in SQLite databases, which provide a secondary target for data recovery. In particular, ostensibly deleted records may persist until the database is explicitly 'vacuumed'; otherwise, they remain recoverable at a later time ([35, 36]).

5 CLOUD FORENSICS

Cloud computing is fast emerging as the primary model for delivering information technology (IT) services to Internet-connected devices. It brings both disruptive challenges for current forensic tools, methods and processes, as well as qualitatively new forensic opportunities. It is not difficult to foresee that, after an intermediate period of adjustment, digital forensics will enter a new period marked by substantially higher levels of automation and will employ much more sophisticated data analytics. Cloud computing environments will greatly facilitate this process, but not before bringing about substantial changes to currently established tools and practices.

5.1 Cloud Basics

Conceptually, cloud-based IT abstracts away the physical compute and communication infrastructure, and allows customers to rent as much compute capacity as needed. Cloud systems have five essential characteristics: on-demand self service, broad network access, resource pooling, rapid elasticity, and measured service. [37]

The cloud is enabled by a number of technological developments, but its adoption is driven primarily by business considerations, which drive changes to how organisations and individuals use IT services. Accordingly, it also changes how software is developed, maintained and delivered to its customers. Cloud computing services are commonly classified into one of three canonical models – Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). In actual deployments, the distinctions can be blurred and many cloud deployments (and potential investigative targets) incorporate elements of all of these.

The differences between the models are best understood when we consider the virtualised computing environments as a stack of layers: hardware such as storage, and networking; virtualisation, consisting of a hypervisor allowing the installation and lifecycle management of virtual machines; operating system, installed on each virtual machine; middleware and runtime environment; and application and data.

Each of the cloud models splits the responsibility between the client and the Cloud Service Provider (CSP) at different levels in the stack (Figure 3). In a private (cloud) deployment, the entire stack is hosted by the owner and the overall forensic picture is very similar to the

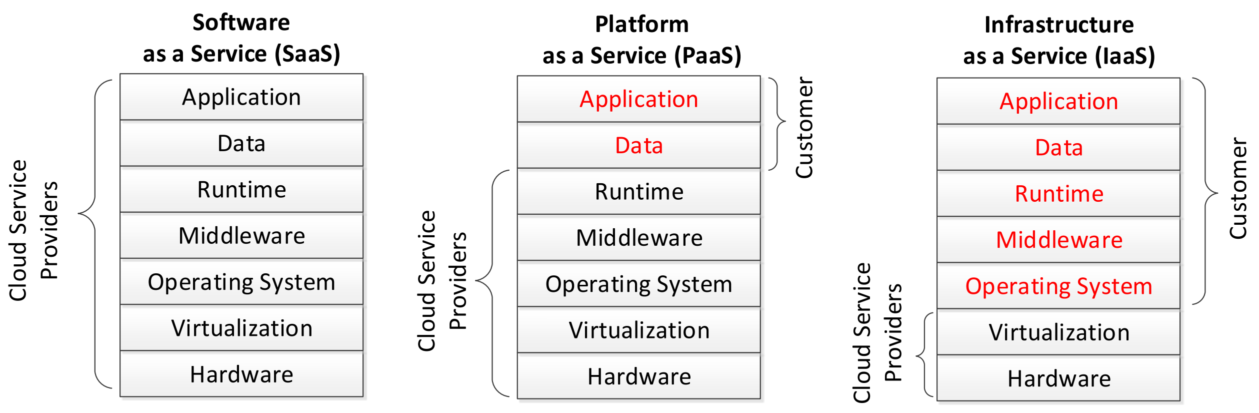

Figure 3: Layers of cloud computing environment owned by customer and cloud service provider on three service models: IaaS, PaaS, and SaaS (public cloud).

problem of investigating a non-cloud IT target. Data ownership is clear, as is the legal and procedural path to obtain it; indeed, the very use of the term 'cloud' in this situation is not particularly significant to a forensic inquiry.

In a public deployment, the SaaS/PaaS/IaaS classification becomes important, as it indicates the ownership of data and service responsibilities. Figure 3 shows the typical ownership of layers by customer and service providers under different service models. In hybrid deployments, layer ownership can be split between the customer and the provider, and/or across multiple providers. Further, it can change over time, as, for example, the customer may handle the base load on private infrastructure, but burst into the public cloud to handle peak demand, or system failures.

5.2 Forensic Challenges

The main technical challenges to established forensic practices can be summarised as follows.

Logical acquisition is the norm. The existing forensic toolset is almost exclusively built to work with the leftover artifacts of prior computations. It relies on algorithmic knowledge of different OS subsystems such as the filesystem in order to interpret the physical layout of the data as acquired from the device.

Physical acquisition is almost completely inapplicable to the cloud, where data moves, resources are shared and ownership and jurisdictional issues can be complicated. Cloud service APIs are emerging as the primary new interface through which data acquisition is being performed.

The cloud is the authoritative data source. Another important reason to query cloud services for relevant information is that they store the primary historical record of computations and interactions with users. Most residual information on the client, such as a cloud drive is transient and often of uncertain provenance.

Logging is pervasive. Cloud-based software is developed and organised differently. Instead of one monolithic piece of code, the application logic is decomposed into several layers and modules that interact with each other over well-defined service interfaces. Once the software components and their communication are formalised, it becomes easy to organise extensive logging of every aspect of the system. Indeed, it becomes critical to have this information just to be able to debug, test and monitor cloud applications and services.

These developments point to logs (of user and system activities) becoming the primary source of forensic information. The immediate implication is that much more will be explicitly known – as opposed to deduced – about the historical state of applications and artifacts. This will require a new set of data analytics tools and will completely transform the way forensic investigations are performed. It will also bring new challenges in terms of long-term case data preservation.

Distributed computations are the norm. The key attribute of the client/standalone model is that practically all computations take place on the device itself. Applications are monolithic, self-contained pieces of code that have immediate access to user input and consume it instantly with (almost) no traces left behind. Since a large part of forensics comprises attributing the observed state of a system to user-triggered events, forensic research and development has relentlessly focused on two driving problems – discovering every last piece of log/timestamp information, and extracting every last bit of discarded data left behind by applications or the operating system.

The cloud model, particularly SaaS, completely breaks with this approach – the computation is split between the client and the server, with the latter performing the heavy computational lifting and the former performing predominantly user interaction functions. Code and data are downloaded on demand and have no persistent place with regard to the client. The direct consequence is that the vast majority of the established forensic tool chain becomes irrelevant, which points to a clear need for a different approach.

5.3 SaaS Forensics

The software industry's traditional delivery model is Software as a Product (SaaP); that is, software acquired like any physical product and is installed by the owner on a specific machine, where all the computations are performed. As a result, the traditional analytical model of digital forensics is physical device-centric – the investigator works with physical evidence carriers such as storage media or integrated compute devices (e.g., smartphones). On the client (or standalone) device, it is easy to identify where the computations are performed and where the results/traces are stored. The new software delivery model – Software as a Service (SaaS) – is subscription-based and did not start becoming practical until the widespread adoption of fast broadband access some ten to fifteen years ago.

The cloud renders many device-centric methods — especially those focused on low-level physical acquisition and analysis — irrelevent. It also requires the development of new tools that can work in the new deployment environment, where the code execution is split between the server and the client devices, the primary storage interface is a service API and the application artifacts are not persistently stored on the device (although local storage may be used as a cache).

Case Study: Cloud Drive Acquisition. Cloud drive services, such as *Dropbox*, *Google Drive* and *Microsoft OneDrive* are the SaaS version of the local storage device, which is central to modern digital forensics. The problem of cloud drive acquisition, a clear first investigative step, is a good illustration of the challenges and opportunities offered by SaaS with respect to forensics.

At first, it may appear that simply copying a local replica of the drive's content is a simple and effective solution. However, this approach offers no guarantees with respect to the accuracy and completeness of the acquisition. Specifically, there are three major concerns:

Partial replication. The most obvious problem is that there is no guarantee that any of the clients attached to an account will have a complete copy of the (cloud) drive's content. As data accumulates online, it quickly becomes impractical to keep full replicas on every device; indeed, it is likely that most users will have *no* device with a complete copy of the data. Furthermore, the acquisition tool needs direct access to the cloud drive's metadata to ascertain its contents; without this information, the acquisition is of an unknown quality, subject to potentially stale and omitted data.

Revision acquisition. Most drive services provide some form of revision history; the look-back period varies, but this is a standard feature that users expect, especially in paid services. Although there are some analogous data sources in traditional forensics, such as archival versions of important OS data structures, the volume and granularity of the revision information in cloud application are qualitatively and quantitatively different. Revisions reside in the cloud and clients rarely have anything but the most recent version in their cache; a client-side acquisition will clearly miss prior revisions, and does not even have the means to identify these omissions.

Cloud-native artifacts. The mass movement towards web-based applications means that forensics needs to learn how to deal with a new problem – digital artifacts that have no serialised representation in the local filesystem. For example, Google Docs documents are stored locally as a link to the document which can only be edited via a web app. Acquiring an opaque link without the actual content of the document has minimal forensic utility. Most services provide the means to export the web app artifact in a standard format such as PDF; however, this can only be accomplished by requesting directly from the service (manually or via an API).

In summary, bringing the traditional client-side approach to drive acquisition to bear on SaaS acquisition has major *conceptual* flaws that are beyond remediation; a new approach is needed, one that obtains the data directly from the cloud service.

6 ARTIFACT ANALYSIS

[38, 39, 40, 41, 42]

Once the external (serialised) representation of a digital artifact such as a text document or an image is standardised, it provides a convenient level of abstraction, thus allowing the development of artifact-centric forensic techniques.

6.1 Finding a Known Data Object: Cryptographic Hashing

The lowest common denominator for all digital artifacts is to consider them as a sequence of bits/bytes without trying to parse, or assign any semantics to them. Despite this low level of abstraction, some crucial problems can be addressed, the most important one being to identify known content, usually files.

Cryptographic hashing is the first tool of choice when investigating any case; it provides a basic means of validating data integrity and identifying known artifacts. Recall that a hash function takes an arbitrary string of binary data and produces a number, often referred to as a digest, within a predefined range. Ideally, given a set of different inputs, the hash function will map them onto different outputs.

Hash functions are *collision-resistant* if it is computationally infeasible to find two different inputs for which the output is the same. Cryptographic hash functions such as MD5, RIPEMD-160, SHA-1, SHA-2 and the current NIST standard SHA-3[38], are designed to be collision-resistant and produce large 128- to 512-bit results.[1] Since the probability that hashing two different data objects will produce the same digest by chance is astronomically small, we can safely assume that, *if two objects have the same crypto digest, then the objects themselves are identical*.

Current practice is to apply a crypto hash function either to an entire target (drive, partition etc.) or to individual files. The former is used to validate the integrity of the forensic target by comparing before-and-after results at important points in the investigation (e.g., to demonstrate that the integrity of the evidence throughout the chain of custody) whereas the latter are used to work with known files. This involves either removing from consideration common files such as OS and application installations or pinpointing known files of interest such as malware and contraband. The US National Institute of Standards and Technology (NIST) maintains the National Software Reference Library (NSRL) [39], which covers the most common operating system installation and application packages. Other organisations and commercial vendors of digital forensic tools provide additional hash sets of other known data.

From a performance and efficiency perspective, hash-based file filtering is very attractive – using a 20-byte SHA-1 hash, the representation of 50 million files takes only 1 GB. This makes it possible to load a reference set of that size in the main memory and filter out, on the fly, any known files in the set as data is read from a forensic target.

6.2 Block-Level Analysis

In addition to whole files, investigators are often interested in discovering known file remnants, such as those produced when a file is marked as deleted and subsequently partially overwritten. One routinely used method to address this problem is to increase the granularity of the hashes by splitting the files into fixed-size blocks and storing the hash for each individual block. The block size is commonly set to 4 KiB to match the minimum allocation unit used by most operating systems' installations. Given a block-based reference set, a forensic target (RAM capture or disk image) can be treated as a sequence of blocks that can be read block by block, hashed and compared to the reference set.

In this context, we say that a block is *distinct*, if the probability that its exact content arises by chance more than once is vanishingly small. If we knew for a fact that a specific block

[1]A discussion on the known vulnerabilities of cryptographic hash functions is outside the scope of this text.

was unique and specific to a particular file, then (in terms of evidentiary value) finding it on a forensic target would be almost the same as finding the entire file from which it was derived. In practice, we cannot definitely know the distinctiveness of every possible data block; therefore, we use an approximating assumption based on empirical data:

“If a file is known to have been manufactured using some high-entropy process, and if the blocks of that file are shown to be distinct throughout a large and representative corpus, then those blocks can be treated as if they are distinct.” [40] Perhaps the most common transformation that yields high-entropy data is data compression, which is routinely employed in many common file formats, such as audio/video and office documents.

Apart from the direct use of blocks as trace evidence for the (past or current) presence of known files, block hashes can be used to improve file carving results by excluding every known blocks *before* performing the carving process. This can improve results by reducing gaps and eliminating certain classes of false positive results.

6.3 approximate matching

A natural generalisation of the problem of finding identical data objects is to find *similar* ones. In the context of digital forensics, the accepted umbrella term for similarity-based techniques is Approximate Matching (AM). As per NIST’s definition, ‘approximate matching is a generic term describing any technique designed to identify similarities between two digital artifacts’. [41]

This broad term encompasses methods that can work at different levels of abstraction. At the lowest level, artifacts can be treated as bit strings; at the highest levels, similarity techniques could employ, for example, natural language processing and image recognition methods to provide a level of reasoning that is much closer to that of a human analyst. With regard to the whole spectrum of similarity methods, lower-level ones are more generic and computationally affordable, whereas higher-level ones tend to be more specialised and require considerably more computational resources. Therefore, we would expect a forensic investigation to customise its use of AM techniques based on the goals of the analysis and the target data.

Use Cases. Using a common information retrieval terminology, it is useful to consider two variations of the similarity detection problem: *resemblance* and *containment* [43]. Resemblance queries compare two comparably-sized data objects (peers) and seek to infer how closely related they are. Two common forensic applications include: (a) object similarity detection – correlating artifacts that a person would classify as versions of each other; and (b) cross correlation – correlating objects that share the same components, such as an embedded image.

In the case of containment, we compare artifacts that have a large disparity in terms of size and seek to establish whether a larger one contains (pieces of) a smaller one. Two common variations are *embedded object detection* – establishing whether a smaller object (such as an image) is part of a larger one (such as a PDF document), and *fragment detection* - establishing whether a smaller object is a fragment (such as a network packet or disk block) of a bigger one, such as a file.

The difference between resemblance and containment is case-specific and the same tool may work in both cases. However, it is important for analysts to put the tool results into the correct context and to understand the performance envelope of the tools they are using in order to correctly interpret the results.

Definitions. The notion of similarity is specific to the particular context in which it is used. An approximate matching algorithm works by defining two essential elements – features and a similarity function. Features are the atomic components derived from the artifacts through which the artifacts are compared. Comparing two features yields a binary outcome – zero or one – indicating whether the feature match was successful or not. The set of all the features computed by an algorithm for a given artifact constitutes a feature set. It can be viewed as an approximate representation of the original object for the purposes of matching it with other objects.

The similarity function maps a pair of feature sets to a similarity range; it is increasingly monotonic with respect to the number of matching features. That is, all else being equal,more feature matches yield a higher similarity score.

Classes. It is useful to consider three general classes of approximate matching algorithms. *Bytewise* matching considers the objects it compares to a sequence of bytes, and makes no effort to parse or interpret them. Consequently, the features extracted from the artifact are also byte sequences, and these methods can be applied to any data blob. The utility of the result depends heavily on the encoding of the data. If small changes to the content of the artifact result in small changes to the serialised format (e.g., plain text), then the bytewise similarity tends to correlate well with a person's perception of similarity. Conversely, if a small change can trigger large changes in the output (e.g., compressed data), then the correlation would be substantially weaker.

Syntactic matching relies on parsing the format of an object, potentially using this knowledge to split it into a logical set of features. For example, a zip archive or a PDF document could easily be split into constituent parts without understanding the underlying semantics. The benefit is that this results in a more accurate solution with more precisely interpretable results; the downside is that it is a more specialised solution, requiring additional information to parse different data formats.

Semantic matching (partially) interprets the data content in order to derive semantic features for comparison. Examples include perceptual hashes that can detect visually similar images, and methods of information retrieval and natural language processing that can find similarities in the subject and content of text documents.

Researchers use a variety of terms to name the different approximate matching methods they have developed: *fuzzy hashing* and *similarity hashing* refer to bytewise approximate matching; *perceptual hashing* and *robust hashing* refer to semantic approximate matching techniques.

Bytewise approximate matching algorithms are the most frequently used AM algorithms in forensics; they follow an overall pattern of extracting a feature set and generating a similarity digest, followed by a comparison of the digests. A similarity digest (a.k.a., fingerprint or signature) is a (compressed) representation of the feature set of the target artifact. It often employs hashing and other techniques to minimise the footprint of the set and to facilitate fast comparison.

6.4 Cloud-Native Artifacts

Forensic analysis of cloud systems is still in its early stages of development, but it will quickly grow in importance. One new and promising area is the analysis of cloud(-native) artifacts—data objects that maintain the persistent state of web/SaaS applications. [42] Unlike traditional applications, in which the persistent state takes the form of files in a local file system, web apps download the necessary state on the fly and do not rely on local storage. Recall that a web app's functionality is split between server and client components, and the two communicate via web APIs. From a forensic perspective, the most interesting API calls involve (complete) state transfer; for example, opening a document or loading a prior version, triggers the transfer of its full content. Conceptually, this is analogous to the process of opening and reading the content of a local file by an application installed on a device. The main difference is that cloud artifacts are internal data structures that, unlike a file, are not readily available for analysis.

Cloud artifacts often have a completely different structure from traditional snapshot-centric encoding. For example, internally, Google Docs' documents are represented as the complete history (log) of every editing action performed on it; given valid credentials, this history is available via Google Docs' internal API. It is also possible to obtain a snapshot of the artifact of interest in a standard format such as a PDF, via the public API. However, this is inherently forensically deficient in that it ignores potentially critical information on the evolution of a document over time.

7 CONCLUSION

Digital forensics identifies and reconstructs the relevant sequence of events that has led to a currently observable state of a target IT system or (digital) artifacts. The provenance and integrity of the data source and the scientific grounding of the investigative tools and methods employed are of *primary* importance in determining their admissibility to a court of law's proceedings. Digital forensic analysis is applied to both individual digital artifacts such as files and to complex IT systems comprising multiple components and networked processes.

Following the rapid cloud-based transition from Software as a Product (SaaP) to Software as a Service (SaaS), forensic methods and tools are also in a respective process of transition. One aspect is a change of emphasis from state-centric analysis, which seeks to deduce events and actions by looking at different snapshots and applying knowledge about the system's operations, to log-centric analysis, which employs explicitly collected log entries to infer the sequence of relevant (to the inquiry) events. Another aspect is the transition from the low-level physical acquisition of storage device images to the high-level logical acquisition of (primarily) application artifacts via well-defined cloud service APIs. Some of the most important emerging questions in digital forensics are the analysis of the large variety of IoT devices, which are forecast to increase in number to as many as 125 billion by 2030, and the employment of machine learning/AI in order to automate and scale up forensic processing.

CROSS-REFERENCE OF TOPICS VS REFERENCE MATERIAL

Topics	Cites
1 Definitions and Conceptual Models	[2, 3, 4, 5, 6]
2 Operating System Analysis	[6, 21, 22]
3 Main Memory Forensics	[32]
4 Application Forensics	
5 Cloud Forensics	
6 Artifact Analysis	[38, 39, 40, 41, 42]

REFERENCES

[1] R. Carolina, *The Cyber Security Body of Knowledge*. University of Bristol, 2021, ch. Law & Regulation, version 1.0.2. [Online]. Available: https://www.cybok.org/

[2] E. Casey, *Digital Evidence and Computer Crime: Forensic Science, Computers and the Internet*, 3rd ed. Academic Press, 2011, iSBN: 978-0123742681.

[3] United Kingdom Parliament. (1990) Computer misuse act. [Online]. Available: https://www.legislation.gov.uk/ukpga/1990/18/contents

[4] D. Goodstein, *Reference Manual on Scientific Evidence: Third Edition*. National Academies Press, 2011, ch. How Science Works, pp. 37–54. [Online]. Available: https://www.fjc.gov/sites/default/files/2015/SciMan3D01.pdf

[5] The Law Commission. (2011) Expert evidence in criminal proceedings in England and Wales. [Online]. Available: https://www.lawcom.gov.uk/project/expert-evidence-in-criminal-proceedings/

[6] K. Kent, S. Chevalier, T. Grance, and H. Dang, "Guide to integrating forensic techniques into incident response," National Institute of Standards and Technology, Tech. Rep. Special Publication 800-86, 2006. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-86/SP800-86.pdf

[7] Comprehensive Crime Control Act of 1984. [Online]. Available: https://www.ncjrs.gov/App/publications/Abstract.aspx?id=123365

[8] 18 US Code § 1030 - Fraud and related activity in connection with computers. [Online]. Available: https://www.law.cornell.edu/uscode/text/18/1030

[9] D. Goodstein, *Reference Manual on Scientific Evidence: Third Edition*. National Academies Press, 2011, ch. How Science Works, pp. 37–54. [Online]. Available: https://www.fjc.gov/sites/default/files/2015/SciMan3D01.pdf

[10] Forensic Science Regulator. (2014) Codes of practice and conduct, appendix: Digital forensic services FSR-C-107. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/351220/2014.08.28_FSR-C-107_Digital_forensics.pdf

[11] ISO/IEC. (2017) ISO/IEC 17025 General requirements for the competence of testing and calibration laboratories. [Online]. Available: https://webstore.iec.ch/preview/info_isoiec17025%7Bed3.0%7Den.pdf

[12] G. Palmer. (2001) Report from the first digital forensic research workshop (DFRWS). [Online]. Available: https://www.dfrws.org/sites/default/files/session-files/a_road_map_for_digital_forensic_research.pdf

[13] V. Roussev, "Hashing and data fingerprinting in digital forensics," *IEEE Security Privacy*, vol. 7, no. 2, pp. 49–55, March 2009, DOI: 10.1109/MSP.2009.40.

[14] S. Garfinkel, A. J. Nelson, and J. Young, "A general strategy for differential forensic analysis," in *The Proceedings of the Twelfth Annual Digital Forensic Researcg Conference (DFRWS)*, 2012, pp. S50–S59, DOI: 10.1016/j.diin.2012.05.003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S174228761200028X

[15] P. Pirolli and S. Card, "Sensemaking processes of intelligence analysts and possible leverage points as identified through cognitive task analysis," in *Proceedings of the 2005 International Conference on Intelligence Analysis*, 2005. [Online]. Available: http://researchgate.net/publication/215439203

[16] J. J. Thomas and K. A. Cook, Eds., *Illuminating the path: the research and development agenda for visual analytics*. US Department of Homeland Security, National Visualization and Analytics Center (NVAC), 2005.

[17] E. Patterson, E. Roth, and D. Woods, "Predicting vulnerabilities in computer-supported inferential analysis under data overload," *Cognition, Technology and Work*, vol. 3, no. 4, pp. 224–237, 2001, DOI: 10.1007/s10111-001-8004-y.

[18] P. Pirolli, *Information Foraging Theory: Adaptive Interaction with Information*. Oxford University Press, 2009, iSBN: 978-0195387797.

[19] G. Klein, B. Moon, and R. Hoffman, "Making sense of sensemaking 1: Alternative perspectives," *IEEE Intelligent Systems*, vol. 21, no. 4, pp. 70–73, 2006, DOI: 10.1109/MIS.2006.75.

[20] V. Roussev, C. Quates, and R. Martell, "Real-time digital forensics and triage," *Digital Investigation*, vol. 10, no. 2, pp. 158–167, 2013, DOI: 10.1016/j.diin.2013.02.001. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1742287613000091

[21] D. Farmer and W. Venema, *Forensic Discovery*, 1st ed. Addison-Wesley Professional, 2005, iSBN: 978-0201634976.

[22] B. Carrier, *File System Forensic Analysis*, 1st ed. Addison-Wesley Professional, 2005, ISBN: 978-0321268174.

[23] J. von Neumann, "First draft of a report on the EDVAC," *IEEE Annals of the History of Computing*, vol. 15, no. 4, pp. 27–75, 1993, DOI: 10.1109/85.238389.

[24] S. Willassen, "Forensic analysis of mobile phone internal memory," in *Advances in Digital Forensics*. Springer, 2005, pp. 191–204, DOI: 10.1007/0-387-31163-7_16. [Online]. Available: http://dl.ifip.org/db/conf/ifip11-9/df2005/Willassen05.pdf

[25] IEEE. (1990) IEEE 1149.1-1990 - IEEE standard test access port and boundary-scan architecture. [Online]. Available: https://standards.ieee.org/standard/1149_1-1990.html

[26] NVM Express. (2019) NVM express base specification, revision 1.4. [Online]. Available: https://nvmexpress.org/wp-content/uploads/NVM-Express-1_4-2019.06.10-Ratified.pdf

[27] J. Zaddach, A. Kurmus, D. Balzarotti, E.-O. Blass, A. Francillon, T. Goodspeed, M. Gupta, and I. Koltsidas, "Implementation and implications of a stealth hard-drive backdoor," in *Proceedings of the 29th Annual Computer Security Applications Conference*, ser. ACSAC '13, 2013, pp. 279–288, DOI: 10.1145/2523649.2523661. [Online]. Available: http://doi.acm.org/10.1145/2523649.2523661

[28] NIST. (2015) Computer forensic tool testing program. [Online]. Available: http://www.cftt.nist.gov/

[29] C. Stevens. (2011) Formatting, cloning and duplicating advanced format media. Technical Paper, Revision 1.0. [Online]. Available: http://idema.org/wp-content/plugins/download-monitor/download.php?id=1220

[30] P. Chen, E. Lee, G. Gibson, R. Katz, and D. Patterson, "Raid: High-performance, reliable secondary storage," *ACM Computing Surveys*, vol. 26, no. 2, pp. 145–185, Jun 1994, DOI: 10.1145/176979.176981. [Online]. Available: http://doi.acm.org/10.1145/176979.176981

[31] C. King and T. Vidas, "Empirical analysis of solid state disk data retention when

used with contemporary operating systems," in *Proceedings of the 11th Annual DFRWS Conference. DFRWS'11.*, 2011, pp. S111–S117, DOI: 10.1016/j.diin.2011.05.013. [Online]. Available: http://dfrws.org/2011/proceedings/17-349.pdf

[32] M. H. Ligh, A. Case, J. Levy, and A. Walters, *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory*, 1st ed. Wiley, 2014, iSBN: 978-1118825099.

[33] J. Chow, B. Pfaff, T. Garfinkel, K. Christopher, and M. Rosenblum, "Understanding data lifetime via whole system simulation," in *Proceedings of the USENIX Security Symposium*, 2004, pp. 321–336. [Online]. Available: https://www.usenix.org/legacy/publications/library/proceedings/sec04/tech/chow/chow_html/

[34] J. Solomon, E. Huebner, D. Bem, and M. Szeżynska, "User data persistence in physical memory," *Journal of Digital Investigation*, vol. 4, no. 2, pp. 68–72, 2007, DOI: 10.1016/j.diin.2007.03.002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S174228760700028X

[35] S. Jeon, J. Bang, K. Byun, and S. Lee, "A recovery method of deleted record for SQLite database," *Personal and Ubiquitous Computing*, vol. 16, no. 6, pp. 707–715, 2012, DOI: 10.1007/s00779-011-0428-7.

[36] B. Wu, M. Xu, H. Zhang, J. Xu, Y. Ren, and N. Zheng, *A Recovery Approach for SQLite History Recorders from YAFFS2*. Springer Berlin Heidelberg, 2013, pp. 295–299, DOI: 10.1007/978-3-642-36818-9_30.

[37] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," Special Publication 800-145, NIST, 2011.

[38] NIST. (2015) SHA-3 standard: Permutation-based hash and extendable-output functions. FIPS Publication 202, DOI: 10.6028/NIST.FIPS.202. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf

[39] ——. (2016) National software reference library. [Online]. Available: http://www.nsrl.nist.gov/

[40] S. Garfinkel, A. Nelson, D. White, and V. Roussev, "Using purpose-built functions and block hashes to enable small block and sub-file forensics," in *Proceedings of the Tenth Annual DFRWS Conference. DFRWS'10.*, 2010, DOI: 10.1016/j.diin.2010.05.003. [Online]. Available: http://dfrws.org/2010/proceedings/2010-302.pdf

[41] F. Breitinger, B. Guttman, M. McCarrin, V. Roussev, and D. White, "Approximate matching: Definition and terminology," National Institute of Standards and Technology, Tech. Rep. Special Publication 800-168, 2014. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-168.pdf

[42] V. Roussev and S. McCulley, "Forensic analysis of cloud-native artifacts," in *Proceedings of the Third Annual DFRWS Europe (DFRWS-EU)*, 2016, pp. S104–S113, DOI: 10.1016/j.diin.2016.01.013.

[43] A. Broder, "On the resemblance and containment of documents," in *Compression and Complexity of Sequences*, Jun 1997, pp. 21–29, DOI: 10.1109/SEQUEN.1997.666900.

ACRONYMS

AM	Approximate Matching.
API	Application Programming Interface.
ATA	Advanced Technology Attachment.
CFTT	Computer Forensic Tool Testing.
CPU	Central Processing Unit.
CSP	Cloud Service Provider.
CTA	Cognitive Task Analysis.
DFRWS	Digital Forensics Research Workshop.
HBA	Host Bus Adapter.
HDD	Hard Disk Drive.
IaaS	Infrastructure as a Service.
IoT	Internet of Things.
ISO	International Organisation for Standardization.
NIST	National Institute of Standards and Technology.
NSRL	National Software Reference Library.
OS	Operating System.
PaaS	Platform as a Service.
PCI	Peripheral Component Interconnect.
RAID	Redundant Array of Inexpensive Disks.
RAM	Random Access Memory.
SaaP	Software as a Product.
SaaS	Software as a Service.
SATA	Serial ATA.
SCSI	Small Computer System Interface.
SSD	Solid State Drive.
UAV	Unmanned Aerial Vehicle.
URL	Uniform Resource Locator.
USB	Universal Serial Bus.

INDEX