A Chain of Findings for Digital Investigations

by
Pedro De Souza

Submitted in fulfilment of the requirements for the degree
MSc Computer Science
in the Faculty of
Engineering, Built Environment and Information Technology
University of Pretoria
October, 2013

Acknowledgements

To Professor Olivier, I would like to give a huge thank you for all your help, assistance, guidance and arguments throughout my masters. You have truly helped in making my masters what it is today. Thank you for all the hours spent reading emails, exploring my ideas, and giving feedback on my various avenues I decided to pursue. You are fantastic supervisor and a fantastic person. I hope your expert guidance and assistance may benefit many other students, as it has really helped me in this master's journey.

To Kosie Eloff, I would really like to say thank you for always being willing to help in any way you could, for always providing my work with your insight and for using a large amount of your time to assist me — even though you never had to. You are a great lecturer and a great friend. I appreciate everything you have done for me. I truly hope that anyone who is fortunate to work with you (students included) know what an honour it is.

To Melanie van der Westhuizen, thank you for being so understanding during my masters studies, for staying up late with me and for always caring with all of your beautiful heart. Thank you also for the great coffee, conversation and company throughout my dissertation. I really love you.

To Telkom and Gys Booysen, thank you for this amazing opportunity and for believing in me throughout my studies. I look forward to working for such a great company and using all I have learnt throughout my studies to show my appreciation.

To God, thank you for all you have given me and letting me meet such kind, caring and knowledgeable people along the way. I pray that I may always be so blessed.

Abstract

Digital Forensic investigations play a vital role in our technologically enhanced world, and it may incorporate a number of different types of evidence — ranging from digital to physical. During a Digital Forensics investigation an investigator may formulate a number of hypotheses, and in order to reason objectively about them, an investigator must take into account such evidence in its entirety, relying on multiple sources. When formulating such objective reasoning an investigator must take into account not only inculpatory evidence but also exculpatory evidence and evidence of tampering. In addition, the investigator must factor in the reliability of the evidence used, the potential for error (tool and human based) and they must factor in the certainty with which they can make various claims. By doing so and creating a detailed audit trail of all actions performed by the investigator they can be better prepared against challenges against their work when it is presented. An investigator must also take into account the dynamic aspects of an investigation, such as certain evidence no longer being admissible, and they must continuously factor these aspects into their reasoning, to ensure that their conclusions still hold.

Investigations may draw over a large period of time, and should the relevant information not be captured in detail, it may be lost or forgotten, affecting the reliability of an investigator's findings and affecting future investigators' capability to build on and continue an investigator's work. In this dissertation we investigate whether it is possible to provide a formalised means for capturing and encoding an investigator's reasoning process, in a detailed and structured manner. By this we mean we would like to capture and encode an investigator's hypotheses, their arguments, their conclusions and the certainty with which they can make such claims, as well as the various pieces of evidence (digital and physical) that they use as a foundation for their arguments. We also want to capture the steps an investigator took when formulating these arguments and the steps an investigator took in order to get evidence into its intended form. The capturing of such a detailed reasoning process helps to allow for a more thorough reconstruction of an investigator's finding, further improving the reliability that can be placed in them. By encoding the investigator's reasoning process, an investigator can more easily receive feedback on the impacts that the various dynamic aspects of an investigation have upon their reasoning. In order to achieve these goals, our dissertation presents a model, called the Chain of Findings, allowing investigators to formulate and capture their reasoning process throughout the investigation, using a combination of goal-driven and data-driven approaches. When formulating their reasoning, the model allows investigators to treat evidence, digital and physical, uniformly as building blocks for their arguments and capture detailed information of how and why they serve their role in an investigator's reasoning process. In addition, the Chain of Findings offers a number of other uses and benefits including the training of investigators and Digital Forensic Readiness.

Table of Contents

Chapter 1

An Introduction to the Chain of Findings

Objectives

- To set the scene for the dissertation
- To introduce and discuss the purpose of this dissertation
- To discuss the methodology to be used
- To describe the layout of the dissertation

Man cannot discover new oceans unless he has the courage to lose sight of the shore
Andre Gide

Chapter 1: An Introduction to the Chain of Findings

1.1 Introduction

In today's society there exists a large collection of digitally enhanced devices which play a vital role in our everyday lives. These digitally enhanced devices range from cell phones [32] for simple home use to large scale databases and servers which handle large volumes of users and online transactions [57]. Each of these devices may contain potentially vital information for a digital investigation which can be used as digital evidence in these investigations. These devices may, however, make use of a variety of file formats for storing such data [59]. A correct understanding of these storage formats is required in order to understand their associated syntax, as well as to correctly interpret their intended meaning (semantics) and hence their content. This form of understanding is also essential to be able to identify any abnormalities in the data [59][73]. The growing number of these devices means that there is a mass of such mediums, each containing increasing memory storage capabilities and an increasing number of formats. This all contributes to the rising potential time that is required for an investigation, as well as the complexity of an investigation [69].

Digital evidence is latent by nature and, as a result, investigators have to use a number of tools [4][10][15][54] to correctly interpret the data, and to begin making discoveries for the investigation [73]. Digital evidence is also fragile [4], and may hence be easily altered, damaged or otherwise destroyed if handled improperly [73]. An investigator should therefore take sufficient precautions to preserve the evidence and its integrity, as well as to support the reliability of collected evidence and conclusions drawn from it. One such precaution is the Chain of Custody which requires an investigator to record and keep track of which parties are accountable for evidence at each point in time, from when it is obtained to when it is returned to its rightful owner or otherwise destroyed [68]. Investigator(s) may however use a number of tools in order to image evidence [32], to analyse evidence [47] and to otherwise transform evidence into its required form. The fragile nature of evidence means that an investigator should not only capture a detailed audit trail of all their actions and the rationale behind them, but also that all tools should be thoroughly tested [10] and calibrated [24][39] before being used. The maintenance of a detailed audit trail of these tests and the calibration process helps to support the reliability of the evidence produced by the tool [23], and, hence, the reliability of the conclusions drawn from the evidence [73]. Casey [13] describes how an investigator must factor in the certainty of the evidence used and the conclusions drawn from the evidence.

In order for an investigator to formulate their conclusions, they can pursue an evidence-driven approach, a goal-driven approach or a combination thereof. In the evidence-driven approach an investigator collects and uses evidence in order to formulate various conclusions. A more goal-driven approach is discussed by Ahmad [2], when he describes how

investigations are typically composed of two main phases, namely an exploratory phase (induction) and an evidence phase (deduction). The exploratory phase is where the investigator constructs hypotheses for what they believe happened and who may have done it. The evidence phase then focuses on the collection (and use) of evidence to serve as a form of proof for the hypotheses created in the exploratory phase. These hypotheses therefore help to guide the investigation. The proof of these hypotheses may however not be the same as more formal methods used in scientific approaches. The reasoning being that it may be too difficult due to resource limitations — namely time, processing power, and various funding, particularly as a result of the converging nature of digital space [15].

When investigators form these proofs they must be sure to take evidence into account in its entirety, to ensure they form a more complete picture of the incident and hence draw more sound conclusions [47]. Investigators must therefore take into account not only inculpatory (evidence that supports their hypotheses) but also exculpatory (evidence that refutes their hypotheses) and evidence of tampering [10] when forming their conclusions and they must capture how such evidence was factored into their reasoning to form sound conclusions. An investigator should also explore alternate explanations and show why they cannot hold [13]. An investigator should therefore create various arguments to show and reason why their particular finding holds. Casey [13] states that completely reliable sources of digital evidence do not exist and digital data is circumstantial at best and as a result it is difficult to implicate an individual with digital data alone. An investigator must therefore incorporate confessions and physical evidence [13] into their arguments as well in order to address this issue. Each of the digital devices may contain vital fingerprints and biological evidence [4] which can provide useful clues and information [75] about the incident and who is accountable. The incorporation of not only digital evidence and physical evidence can hence help to provide a more complete picture of the incident, and therefore serves a vital role as building blocks of an investigator's arguments and reasoning. Furthermore, investigators may use existing findings as building blocks for new arguments. In this manner they use existing knowledge to further deduce conclusions and what they know about the investigation. Investigators then make use of reports and presentations in order to communicate their arguments, their reasoning and their findings in the presentation phase of an investigation [47][73].

In the presentation phase, an investigator must include all information about their hypotheses and their logical arguments for or against these hypotheses [58], when communicating their work. Rowlingson [58] states that these arguments must not only be objective but also well-formed. He continues with how an investigator must also capture and clearly show the evidence that supports these arguments. The National Institute of Justice [47] notes that when an investigator presents their findings, how they should also describe how these findings were derived and discovered. They also discuss how an investigator must provide (and include) information about the processes and analysis they performed that helped achieve these findings, as well as all supporting materials for these findings. These supporting materials help to demonstrate the reliability of evidence and include audit trails, testing documentation and the Chains of Custody. While these are important aspects, what of how and why the evidence supports these hypotheses and hence an investigator's findings? Was it that the

evidence took place at a particular point in time (such as CCTV footage), was it that the evidence was encrypted (such as a particular file or a means of communication), or even simply that the evidence existed that allowed it to serve as a building block in an investigator's reasoning?

The capturing of not only evidence, but also the various attributes (or characteristics [57]) of the evidence, as well as the criteria these attributes comply with — allows an investigator to capture more in depth information of how evidence serves its role as a building block in an investigator's arguments (and reasoning). By allowing the investigator to capture why the evidence and its attributes are important for the investigator's reasoning, as well as why they must meet certain criteria — it allows the investigator to capture the rationale behind their building blocks and their overall reasoning. Earlier we discussed how an investigator must factor in the certainty of the evidence they use and the conclusions they draw from the evidence. The evidence and the criteria they meet can be seen as building blocks for the investigator's argument(s) and help increase the certainty of the investigator's arguments. The amount these building blocks contribute will however vary depending on their particular contribution and their reliability. An investigator must also take into account the dynamic aspects of an investigation, such as certain evidence no longer being admissible, and they must continuously factor these aspects into their reasoning to ensure that their conclusions still hold. They should also factor these aspects into the certainty with which they can make their conclusions.

What if we could capture not only the certainty of an investigator's arguments but also the investigator's arguments, the steps (and their sequence) they followed to formulate it, the building blocks (their provenance information and how they were combined) that make the foundation of the argument? What if we could capture the rationale behind their choices, reasoning and the processes they followed? We collectively refer to these aspects as an investigator's reasoning process. What if, in addition, we could provide a means for investigators to more easily receive feedback on the impact that the dynamic aspects of an investigation have upon their reasoning? Overall what if it were possible to create a form of a Chain of Findings that provides these capabilities for investigators and allows them to capture their reasoning process and all associated information. This is the particular problem that our dissertation addresses, but before we discuss our problem statement let us first explore the requirements we believe are necessary for the Chain of Findings and why we believe they are necessary.

1.2 The Requirements for the Chain of Findings

In this section we discuss various requirements we believe are necessary for the Chain of Findings and we provide motivation for them. Our discussion begins with the investigator's arguments.

In order to correctly and sufficiently capture an investigator's arguments, the investigator must be able to record and capture the various combination of evidence and existing findings, which are used as building blocks to form the foundation of their arguments for each of their

findings (and opinions). In addition, the Chain of Findings must allow investigators to integrate and capture inculpatory evidence, exculpatory evidence and evidence of tampering into their arguments and factor in the certainty of their evidence and claims they derive from it. The Chain of Findings must also allow investigators to capture and record alternate explanations and it must allow them show why these alternate explanations cannot hold. Cohen [15] elaborates on how investigators must use multiple confirmations to support their conclusions. The Chain of Findings should therefore provide such functionality and allow investigators to capture the certainty that each of these confirmations contributes to their arguments, and the associated conclusions.

These arguments must incorporate all information of an investigator's reasoning process, allowing the investigator to record, in detail, why a finding holds and can be relied upon, and/or why it does not. The building blocks must therefore contain attribute information and criteria information, to allow the investigator to capture how these building blocks serve their role in the investigator's reasoning. It is important that the investigator be able to treat these building blocks uniformly, whether they are evidence or existing findings. By doing so the investigator is given the freedom to combine building blocks, as is needed, to more correctly and efficiently capture their reasoning process that was followed. While our focus for our dissertation is Digital Forensics and hence digital evidence, physical evidence has its uses and benefits in such investigations. The model must therefore incorporate the means to cater for digital and physical evidence, and allow investigators to use them to form their reasoning. The next particular aspect that we would like to elaborate on is the certainty of an investigator's findings.

Earlier we discussed how an investigator must factor in the certainty of the evidence they use and the conclusions they draw from the evidence. Investigators should also factor in the certainty of existing findings they use and the conclusions they draw from them. In order to cater for this we must allow investigators to factor in and capture the certainty that their building blocks provide their arguments with, as well as the certainty of the conclusions they derive from these aspects. The incorporation of the manner in which the building blocks are combined in the argument, their attributes and their criteria, as well as the certainty they contribute — helps to provide the Chain of Findings with a means to provide feedback on the dynamic aspects of the investigation. An investigator can then easily receive feedback on the impact these aspects have on their reasoning, and the certainty with which they can make certain claims. An investigator should also have the freedom to not have to use such a weight/certainty approach, if the finding can clearly be shown to draw from its building blocks, or if they deem it simply not needed. It is therefore important that the Chain of Findings provides the investigator with such flexibility, and the necessary capabilities to allow the investigator to select and combine these approaches as is needed, in order to sufficiently capture their reasoning process. Furthermore, the Chain of Findings must provide flexibility in the manner in which investigators can formulate their arguments, achieve their desired findings and overall record their reasoning process. In this manner the Chain of Findings must allow investigators to use evidence (data) driven, goal-driven and a combination of these approaches to form their arguments and their reasoning. The Chain of

Findings should also maintain a detailed audit trail of how the building blocks (evidence and findings) came to be in their current form.

An investigator may have to transform evidence (a number of times while preserving the integrity of the source in each case) in order to get it into the correct form, to serve its role as a building block for the investigator's argument. It is therefore important that the Chain of Findings provide the investigator with these capabilities and record these transformations, the testing thereof and the processes the investigator followed to perform them. The Chain of Findings should also support the capturing of non-digital events regarding evidence's provenance, such as collection, storage and transportation. A detailed recording of all such provenance information persistently helps to verify and support the reliability of the evidence and the actions taken by the investigator particularly by aiding in their reconstruction. In this manner the detailed audit trail also helps to support the certainty that an investigator places in the evidence and the certainly in their conclusions they draw from it. The Chain of Findings should also maintain such an audit trail for each of an investigator's findings, to help to provide a detailed history of how their arguments were derived, as well as any modifications that were made. The important role these audit trails serve in an investigation [47] along with the potentially long duration of investigations [68] means that these audit trails should be recorded in a well-defined manner to ensure that they are correctly interpreted.

Non-repudiation plays an important role in these audit trails, both for evidence and an investigator's findings. An investigator is accountable for each of their actions while the evidence is in their custody [32], especially due to the fragile nature of digital evidence [73]. An investigator's reasoning and the conclusions they derive may impact peoples' lives — whether it is as a result of convicting a criminal or showing a party's innocence [13]. The Chain of Findings must therefore integrate non-repudiation into each of its recordings of an investigator's actions and their reasoning, so that an investigator cannot easily deny their involvement in them. While non-repudiation is an important part of our dissertation we will however not be focusing on authentication or access control.

The final component of the reasoning process that must be captured is the rationale of the investigator and the steps that they followed. In order to do so the Chain of Findings must allow the investigator to capture the rationale behind each of the steps of their reasoning process for findings. This means that the investigator must be able to capture the rationale behind each of the building blocks including the transformations done and the criteria that they must comply with. The investigator must then also be able to capture the rationale of why the building blocks were combined in the manner they were and why they form the foundation of the investigator's argument(s) for a finding. An investigator must also be able to record the rationale behind the certainty. While we have emphasized the importance of flexibility in capturing an investigator's reasoning, the Chain of Findings must also be designed with flexibility in mind with regards to its structure and its capabilities. Digital devices and media are growing and evolving at a rapid rate and the Chain of Findings should therefore incorporate flexibility in its design to cater for these advancements and the transformations that their digital contents may require.

1.3 The Problem Statement

In this dissertation our main focus is to determine whether it is possible to create a Chain of Findings which meets the requirements we discussed — providing a formal means to capture and express an investigator's reasoning process, and providing feedback on the impacts of the dynamic aspects of an investigation. For the sake of completeness for our problem statement, we provide a summary of the requirements that we believe are necessary for the Chain of Findings.

The investigator must be able to capture their conclusions, their arguments behind their conclusions and the rationale behind each of the steps of their reasoning process. The Chain of Findings must provide flexibility in the manner in which investigators can formulate their arguments, factor in certainty and achieve their desired findings. The Chain of Findings must capture all details of an investigator's reasoning process, including the combination of building blocks that an investigator uses for their arguments. These building blocks may be evidence — digital or physical — and existing findings. Investigators must be able to use these building blocks uniformly and capture the attributes and criteria that allow these building blocks to serve/not serve their roles in the investigator's reasoning. The Chain of Findings should incorporate the means for investigators to reliably transform their building blocks, as is needed, and record a detailed audit trail of the transformation and its reliability. The Chain of Findings should also allow for the capturing of provenance information regarding not only the transformations performed but also physical events, and it should ensure that non-repudiation is a core aspect of all recorded events. The Chain of Findings must also allow investigators to incorporate multiple confirmations for their reasoning, and it must allow investigators to capture alternate explanations and show why they cannot hold. Finally the Chain of Findings should be designed with flexibility in mind to allow it to grow and adapt with the advancements in digital devices and media.

Our focus for this dissertation is on Digital Forensics, and hence digital evidence. It is also important for us to note that our dissertation is not focusing on or addressing authentication and access control. Now that we have discussed our problem statement let us explain the methodology we will be using to achieve our goal.

1.4 Methodology

The main methods that are used in this dissertation to achieve the Chain of Findings is a model and a prototype. The model is the primary method that is used due to its capability to simplify reality [25]. It allows us to focus purely on the aspects of interest to the problem and ignore the others [56]. Our secondary method is a prototype.

The prototype serves as a proof of concept for the Chain of Findings, showing that the Chain of Findings is indeed possible, and not purely theoretical. The prototype allows us to discuss the design and implementation issues that we encountered, as well as lessons that we learned during the implementation process. The prototype is implemented using Java, as it is freely available and it is a language that the author is well grounded in.

In the remaining two sections of Chapter 1, we discuss the terminology that is used in this dissertation (section 1.5), followed by the layout of the dissertation (section 1.6).

1.5 Terminology

We discuss the terminology to elaborate on what is meant by these terms and to help avoid any misunderstandings or misinterpretations when they are used. For each of the terms discussed in this section, we first introduce the term, followed by its definition in this dissertation's context and finally how and why it will be used in the dissertation.

- **Digital Evidence**

Digital evidence is information and data that has investigative value, is stored on or transmitted by an electronic device, is latent in nature, fragile and may be time sensitive [4]. Digital evidence is the focus of our work and its fragile nature plays an important role in the various steps we take to achieve our Chain of Findings.

- **Digital Forensic Readiness**

Digital Forensic Readiness is when an organization has the necessary means and infrastructure in place to collect and process digital evidence should an incident occur [64]. The means and infrastructures include security policies, procedures, practices, mechanisms and training programs [24] that allow for a proactive approach to an investigation. The proactive approach aims to minimize the cost, labour and time required for an investigation compared to more reactive ones. Digital Forensic Readiness aims to reduce the interruption to the organization [58] should an investigation be required and overall it helps to maximize the capability of an organization to have more sound and credible approaches for capturing, processing and handling of digital evidence [24][58][64]. Digital Forensic Readiness is an important goal for an organization, and the capability for the Chain of Findings to help achieve such a goal (Chapter 6) serves an important motivation for pursuing such a model.

- **Digital Forensics**

Digital Forensics is the process, techniques and tools [10] for discovering, identifying, collecting [73] and retrieving [51], preserving, recovering, examining, interpreting, documenting and presenting digital evidence in a forensically sound manner, using accepted [32], scientifically derived and proven methods [11] in a manner that is legally acceptable [8][43]. It includes the following of all steps and precautions in order to comply with the legal requirements during all phases of an investigation [43], preserving the integrity of evidence throughout the investigation so that any evidence attained is admissible in the court of law [8][74] and that any bias throughout the investigation has been avoided [17]. It focuses on not only determining what happened in an incident, but also on redress — namely holding parties accountable for their actions [24]. Digital Forensics is the focus of this dissertation.

- **Effectiveness**

Effectiveness refers to how good an entity is at doing at what it is intended to do and what it is designed to do [61]. This entity may be a system, component, person or other form of an object in the real world. Effectiveness is an important concern for the Chain of Findings, as the Chain of Findings must be good at capturing and recording all the relevant information, in order for it to be of use to investigators and the Digital Forensics field in general.

- **Efficiency**

Efficiency refers to the manner in which the system supports the user in carrying out their tasks [61]. Efficiency is important for the Chain of Findings, as investigators may have a large volume of evidence to analyse, from which they must formulate their findings. An investigation is a complex and time consuming task and, as a result, it is important that the Chain of Findings incorporate all the necessary capabilities that are required in order to assist the investigator in capturing all the relevant information. The Chain of Findings must hence be efficient by providing all the necessary capabilities and it should allow the investigator to capture the relevant information, without placing further, unnecessary burden on the investigator.

- **Evidence**

Digital data (and hence digital evidence) are circumstantial and they must be combined with confessions, video surveillance and physical evidence to overcome uncertainty [13]. We collectively refer to digital evidence and physical evidence as evidence for ease of reference, and to simplify our discussions. The focus of our dissertation however is on digital evidence but, due to the important role physical evidence plays in an investigation involving digital evidence [2], we must cater for it as well. An investigator may use a combination of these forms of evidence in order to formulate arguments, and catering for only digital evidence may reduce the applicability of our model.

- **The Fragile Nature of Digital Evidence**

The fragile nature of digital evidence refers to its capability to be easily altered, damaged or otherwise destroyed if it is improperly handled [73]. The fragile nature of digital evidence means that sufficient steps must be taken to preserve the evidence and its integrity and is hence an important reason for pursuing non-repudiation and the Proof of Action (Chapter 3) in our dissertation.

- **Investigator**

An investigator in this dissertation is a generic term given to any party that may be involved in an investigation, and/or the reasoning process behind the conclusions of such an investigation. There are number of parties involved in an investigation (introduced in Chapter 2), each of which may perform a number of distinct roles and actions [73]. We believe such a generic term is beneficial in order to avoid discussions on these distinctions — which are not the focus of this dissertation — and to simplify our discussions.

- **Metaphor**

A metaphor is a tool used to explain to a user how to understand a particular entity or concept and how it can be used [61]. The metaphor does this by referring to and describing something that the user is familiar with and understands. A metaphor hence capitalises on a user's existing knowledge, and uses it to explain something that they are unfamiliar with, something that is new, and/or something that may be difficult to understand. The metaphor encourages the user to reason about how the metaphor's behaviours and properties apply to the complex entity or concept, once a proper understanding is achieved. The metaphor also provides an easy way for a person to refer to this (complex) entity or concept. Metaphors play an important role in this dissertation, helping to discuss components in the Chain of Findings, and also to make it easier to refer to these components.

- **Model**

A model is an abstraction or simplification of reality, allowing us to focus only on the aspects of interest, or more appropriately put, the aspects which are important to the particular domain of interest [57]. We use a model to describe the Chain of Findings.

- **Precondition and Postcondition**

A precondition is a particular condition that must be met in order to ensure the correct execution of a sequence of steps or algorithm(s) that may follow after the condition [21][31]. Preconditions are well known terms in programming and they can be seen as effective metaphors for describing how evidence's and finding's attributes must meet certain criteria (preconditions), before they are able to play their role as building blocks. They also make it easy to refer to the property meeting such criteria without having to continuously restate the more verbose form.

A postcondition is similarly defined and it refers to a particular condition that must be met after the execution of a sequence of steps or algorithm(s). We make use of postconditions to refer to the various conditions that should hold after a tool's execution and the various conditions that must be met should an investigator's findings hold.

- **Pseudocode**

Pseudocode is a notational system that allows one to express ideas and algorithms in a more natural language than a programming language [7]. It makes use of well-defined textual structures (which resemble structures in programming languages) but still allows one to avoid the complexity and syntactic rules of a programming language. We will be using pseudocode in order to describe certain algorithms and approaches in this dissertation. This will allow a wider audience to be able to understand the algorithms, rather than only those familiar with a specific programming language. Pseudocode also closely resembles structures used in programming languages, which makes it easier to convert the pseudocode into a specific programming language. The main benefit is for readers who are interested in the implementation of certain components.

1.6 Dissertation Layout

In order to discuss the dissertation layout we have provided a brief overview of each of the chapters and their contents. We then end section 1.6 with a diagram which provides a visual map of our dissertation's layout and illustrates the relationships between chapters.

- **Chapter 2: Digital Forensics and an Investigator's Reasoning Process**

Chapter 2 provides a background behind the need, purpose and importance of the Chain of Findings, and the essential role that Chapters 3 and 4 serve in achieving the Chain of Findings (Chapter 5). In this chapter we discuss what is meant by Digital Forensics and digital evidence, and we elaborate on the various phases of an investigation. In these phases we discuss the fragile nature of evidence, the large number of tools that an investigator requires and how these tools as well as an investigator's actions, can affect the integrity of evidence. Throughout these phases we also elaborate on an investigator's reasoning process. Our discussion on an investigator's reasoning allows us to discuss the various difficulties of the reasoning process and existing approaches for performing such reasoning. It also allows us to explain the important role that an investigator's reasoning has on the investigation and the investigation's outcome.

- **Chapter 3: Non-Repudiation and an Investigator's ID**

An investigator is accountable not only for the actions they take while evidence is in their custody [32] (particularly due to its fragile nature) but also for the reasoning and conclusions they derive, which may impact people's lives [13]. Chapter 3 allows us to discuss how we can create a type of a proof of action for each of an investigator's actions during their reasoning process, which cannot be easily denied or fabricated. In Chapter 4 we then use these Proof of Actions to create a reliable audit trail of all actions relating to/involving evidence and findings (which is further discussed in Chapter 5). Chapter 3's relationship with Chapter 4 and 5 can be seen by the arrows in Figure 1: The Layout of the Dissertation.

- **Chapter 4: Containers and the Transformation Manager**

An investigator may require a large number of tools in order to correctly capture evidence and transform it into a suitable form for it to serve a role in their reasoning process. These tools play an important role in the reasoning process, and their correctness is essential for ensuring the reliability of an investigator's reasoning. An investigator must therefore maintain an audit trail of all actions they perform [4][73] and thoroughly test all tools they use. Unfortunately these are time-consuming tasks, and investigations have limited resources. Detailed audit trails and testing may therefore suffer as a result. This highlights the need for a particular system to manage these tools, to test them and to ensure that they record their actions accurately. We call such a system the Transformation System. In this chapter we also elaborate on how an investigator uses evidence (digital and physical) and findings they make as building blocks in order to formulate further findings. These building blocks may however have a large amount of information regarding them (such as metadata) and how they came to be (provenance information). In order to help capture all such information in an easily locatable area we introduce two types of containers, namely Evidence and Finding Containers. Evidence Containers are discussed in detail in this

chapter, whereas Finding Containers are left for Chapter 5 to provide greater detail. We then discuss how the Transformation System can be used to create, configure and transform these Containers, in a well-documented and reliable manner.

- **Chapter 5: The Chain of Findings**

In Chapter 5 we explore the Finding Container further, discussing its structure and its purpose. We elaborate on how these Finding Containers can be used by an investigator to capture their findings, the rationale behind them, the certainty with which they can make these claims and the reasoning behind their findings. We explain how such a Container allows an investigator to formulate and capture their reasoning by factoring in inculpatory evidence, exculpatory evidence and evidence of tampering through the use of Evidence Containers. We also elaborate on how the Finding Container allows an investigator to capture alternate explanations and how the Finding Container can capture how these alternate explanations were reasoned about, and shown not to hold. We continue with how such a Container allows an investigator to receive feedback on the dynamic aspects of an investigation and the impacts these aspects have upon their reasoning and the certainty with which they can make certain claims. Furthermore, we elaborate on how an investigator can use not only Evidence Containers but also Finding Containers as building blocks to form their reasoning — allowing existing findings to be used as a basis for further findings. Once our discussion on the Finding Container is complete we then discuss the Chain of Findings.

The Chain of Findings is the heart of our dissertation, and it is a model which incorporates all the components we have discussed up to this point in the dissertation to provide the means for investigators' to accurately manage, capture and record their reasoning process. This includes the Containers, Evidence and Findings based, and their creation and transformations.

- **Chapter 6: A Discussion on the Model and an Example of its Usage**

Chapter 6 provides a more in-depth discussion on the Chain of Findings presented in Chapter 5, focusing on the benefits and drawbacks of the model. We then provide a real world example to demonstrate and further explore the aspects discussed in the chapter.

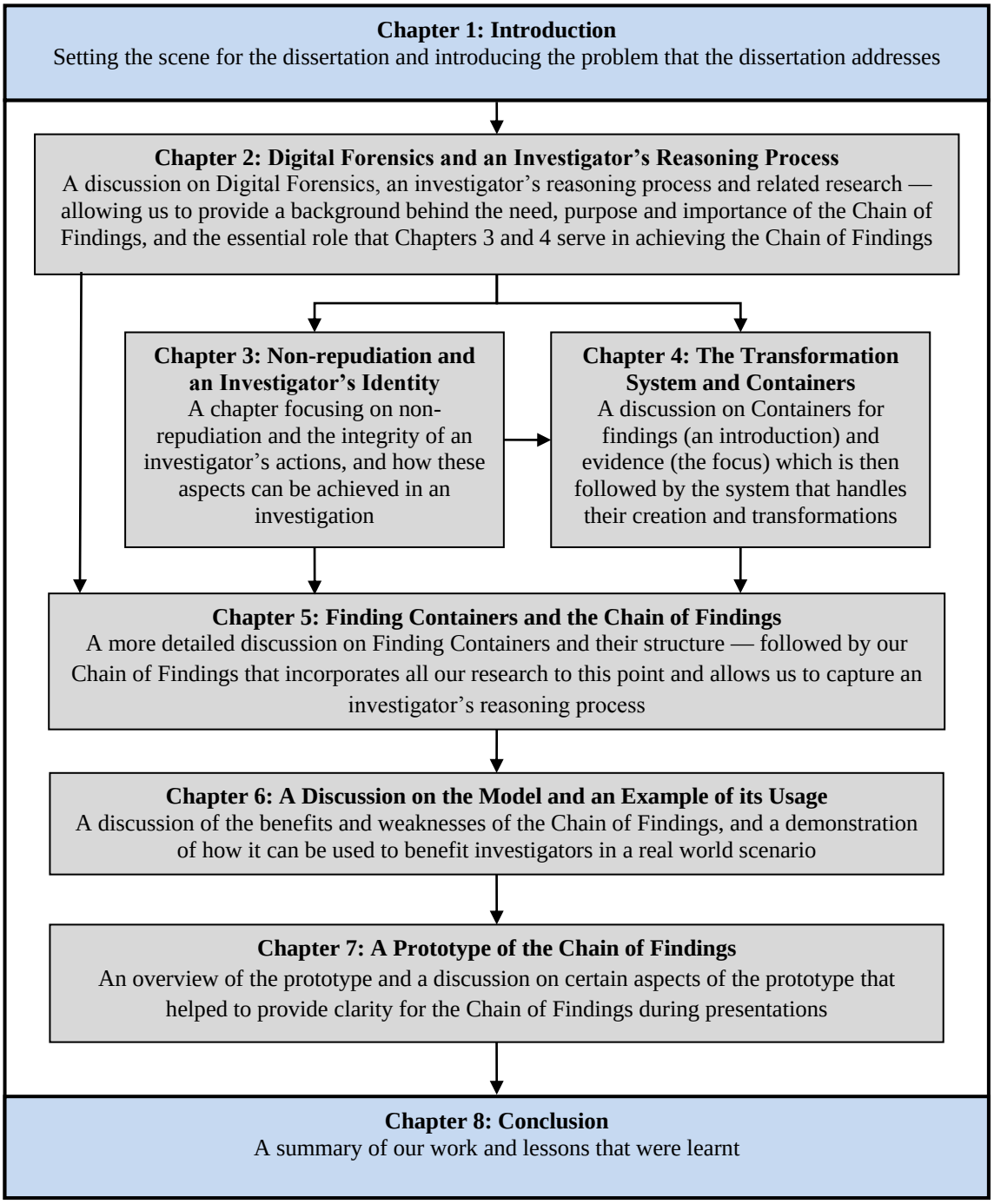
- **Chapter 7: A Prototype of the Chain of Findings**

In Chapter 7 we discuss our proof of concept for the Chain of Findings, where the proof of concept is a prototype coded using Java. We use this chapter to discuss the design choices that were made and the lessons that were learnt from implementing the Chain of Findings.

- **Chapter 8: Conclusion**

In Chapter 8 we provide a summary of our work and the lessons that were learnt — allowing us to recapitulate on the overall problem that was addressed in our work and on our model that was used to address the problem.

Figure 1: The Layout of the Dissertation



1.7 Chapter Summary

Chapter 1 has allowed us to set the scene for the dissertation and explain our focus for this dissertation, namely the Chain of Findings. In this chapter we have elaborated on the various benefits of such a model and we have discussed the various facets it should incorporate. We have also discussed the terminology that we use in this dissertation, to help avoid ambiguity

and misinterpretations. Furthermore we provided a layout of our dissertation to explain our master's journey to the reader. Let us now begin our journey with Chapter 2 Digital Forensics and an Investigator's Reasoning Process.

Chapter 2

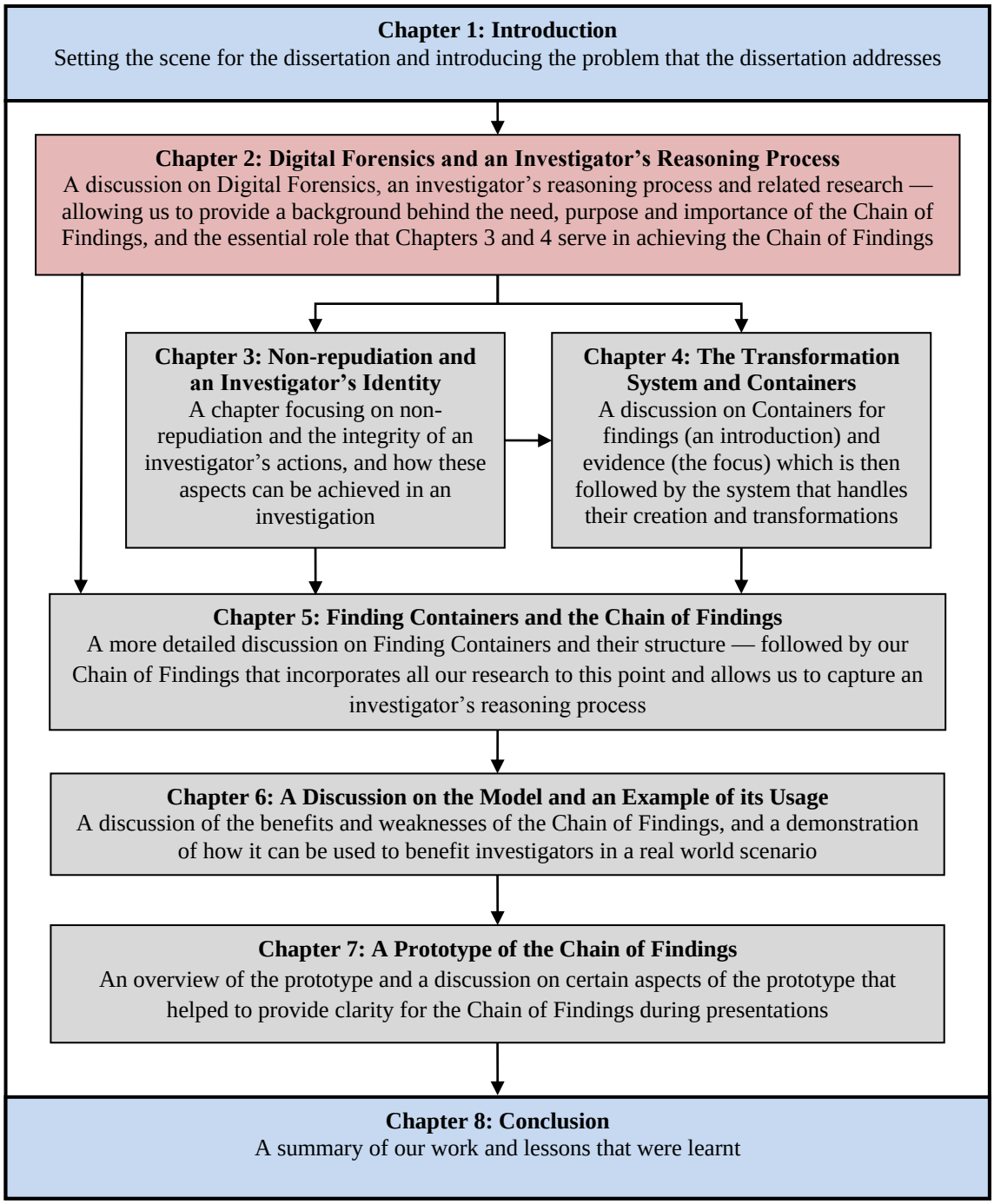
Digital Forensics and an Investigator's Reasoning Process

Objectives

- To investigate peer reviewed research relating to Digital Forensics and an investigator's reasoning process
- To discuss, summarise and learn from such research, and incorporate its contained knowledge into the paper
- To establish terminology

Do not seek to follow in the footsteps of the men of old; seek what they sought
Basho

A Map of our Current Location in the Dissertation



Chapter 2: Digital Forensics and an Investigator's Reasoning Process

2.1 Introduction

This chapter provides a background for the upcoming chapters and the dissertation as a whole. In this chapter we discuss Digital Forensics, investigations and evidence. Our discussions on these aspects allow us to elaborate on the particular drawbacks of digital evidence and why an investigator is required to formulate arguments. It allows us to discuss inculpatory evidence, exculpatory evidence and evidence of tampering, as well as discuss how an investigator must factor in these types of evidence in order to form sound arguments. We also elaborate on the potential approaches an investigator may take in order to form such arguments. In this way we can provide a background behind the importance of capturing an investigator's reasoning process and hence our dissertation as a whole. The discussion on Digital Forensics allows us to elaborate on what is meant by the term in this dissertation and it allows us to elaborate on the various phases of a Digital Forensic investigation. Each of these phases can have potential impacts on the evidence and, hence, any reasoning based upon these building blocks.

Our discussion on these potential impacts, the emphasis on the fragile nature of evidence and the importance of taking the necessary precautions, helps make apparent the potential drawbacks and ramifications on the investigator's conclusions should they not be adhered to. In this way we emphasize how an investigator is accountable for each of their actions. Our discussion allows us to provide a rationale for the need of a Proof of Action (Chapter 3) which associates an investigator with the actions they perform, to providing non-repudiation. We also discuss the various tools (and transformations) an investigator may require throughout the investigation in order to interpret and correctly transform evidence into its required form, such that an investigator can formulate their reasoning and draw their conclusions. The potential impacts of these tools on evidence (and its integrity) mean that there is a strong requirement for testing these tools, and the maintenance of a detailed audit trail of all of an investigator's actions [73]. This allows us to provide a background for the Transformation System (Chapter 4) which facilitates the transformation of evidence into its required form as well as the testing and documentation of the tools involved. In this manner the Transformation System can help provide reliability to the transformed evidence, and hence the arguments which are built from these building blocks. One phase that is of particular interest to this dissertation is the analysis phase.

The analysis phase involves placing evidence into a logical and useful format, and building arguments and drawing conclusions from the evidence. Once our discussion on the phases of an investigation (section 2.3) is completed we then elaborate more on the analysis phase and on the literature related to the linking and correlation of evidence and events. These sources elaborate on how such linking and correlation must be used in order for an investigator to

draw conclusions and to support their reasoning. A particular, well-known means for reasoning from facts and drawing conclusions, is an expert system and it helps to provide a generally well known background for integrating this various literature, and explaining an investigator's reasoning from facts.

Expert systems and their various components, including a fact database, a set of rules, and an explanation system are then used as metaphors to explain an investigator's reasoning (section 2.4). Expert systems use the facts that they know about their environment in order to perform actions and draw conclusions (with a particular certainty), similar to how an investigator uses evidence from a particular incident in order to reason and draw conclusions in an investigation, with a particular certainty. The explanation facility of an expert system allows it to explain to a user why certain actions and conclusions were made. An investigator must incorporate such explanations into their arguments, however these explanations take the form of detailed audit trails and reports, which capture all actions and steps that the investigator performed in an investigation [73] and which also explain how they came to the conclusions they did [58]. While expert systems are not directly related to Digital Forensics, they provide an efficient means for us to relate and integrate various Digital Forensic sources with regard to reasoning in an investigation and the certainty an investigator can place into their results, in a more coherent manner.

Before we continue with the body of the chapter it is important to note that there may be a large number of personnel and parties involved in an investigation. These involve first responders who focus on securing the scene and the preservation of evidence (digital and physical) [32] evidence collection staff, evidence recovery staff, and external consulting agencies that have specialized skills that may be required in an investigation [73]. Other parties involved in a digital forensic investigation include law enforcement, lawyers [41], incident response teams, administrators [24], forensic analysts, evidence custodians [32], management, technical staff [75] and expert witnesses [15]. In order to simplify our discussions in this dissertation, as well as to avoid discrepancies and having to explicitly specify exact details for each role, we collectively refer to these parties as investigators. While investigators are however referred to as parties who play important roles in the planning and management of the investigation [32], and in the identification and presentation of digital evidence and its meanings [73], the use of a collective word simplifies our discussions for the chapter and the dissertation as a whole.

2.2 Digital Forensics and Reasoning in an Investigation

Investigations may take place as a result of internal incidents [2][8] — such as racism, harassment, fraud, computer misuse (gambling, playing of games, pornography) [4] — or external incidents to an organization — such as intrusions [71], DDOS and malware attacks [73]. Further incidents include murders, fraud, narcotics, child pornography and piracy [4]. In each if these situations there is a violation of some form, whether it is the violation of a security policy [76], and/or of ethics and correct behaviour as defined by an individual, organization and/or by law. In each of these incidents an investigation may be required to

determine whether the suspect was indeed involved and whether disciplinary action, and/or legal action [11] may be required. Investigations can also play an important role in insurance claims should an incident occur, allowing one to assess the (potential) damage to a system as a result of an incident [58].

Our use and dependence on technology [17], and the powerful capabilities of these devices means that these digital devices may play important roles in these incidents (and investigations due to these incidents) for a number of reasons [4]. Digital devices may have been used to commit the crime and/or they may contain vital information about the suspect, the crime and the actions performed using the device [43]. Our reliance on these digital devices and hence the valuable information they may contain, means that they may also be the target of an incident [17], such as in DDOS or fraud. Our increasing reliance on these rapidly developed devices [42] means that the range and number of scenarios and incidents in which these digital devices, and the valuable data they may contain, may be required in an investigation, increases as well [71]. The reliability and integrity of this data is essential because of the important role that the data these devices may potentially contain can play in convicting a criminal as well as showing a party's innocence [13].

Digital Forensics is a broad and maturing field [10] and incorporates all digital devices which may contain vital information for an investigation. It includes a number of more specialized fields including Computer and Network Forensics [72][75], Database Forensics, File System Forensics [51] and Mobile Forensics [32] — but what exactly is Digital Forensics? Digital Forensics is the processes, techniques and tools [10] for discovering, identifying, collecting [73] and retrieving [51], preserving, recovering, examining, interpreting, documenting and presenting of digital evidence in a forensically sound manner, using accepted [32], scientifically derived and proven methods [11] in a manner that is legally acceptable [8][43]. It includes the following of all steps and precautions in order to comply with the legal requirements during all phases of an investigation [43], preserving the integrity of evidence throughout the investigation so that any evidence attained is admissible in the court of law [8][74] and that any bias throughout the investigation has been avoided [17]. Farmer and Vemena (discussed in [8]) and the Digital Forensics Research Workshop (discussed in [11]) go on to state that the purpose of Digital Forensics is to facilitate or further the reconstruction of events and data in an investigation.

Traditionally, security focused on the resistance of attacks, the recognition of attacks and the recovery of attacks but we are now moving towards redress [24]. Redress refers to our capability to hold parties accountable for their actions. Sommers describes how there is a distinction between Digital forensics (lowercase 'f') and Digital Forensics (uppercase 'f') [24]. Digital forensics (lowercase 'f') focuses on determining what happened, how an incident occurred and on the restoration of systems affected by an incident. Digital Forensics on the other hand includes the steps involved to determine and hold the correct parties accountable for these incidents (and their actions). In this dissertation we refer to the latter, namely Digital Forensics (uppercase 'f) and the importance of digital evidence.

Digital evidence is information and data that has investigative value, and that is stored on or transmitted by an electronic device, is latent in nature, fragile and may be time sensitive [4]. The fragile nature of digital evidence refers to its capability to be easily altered, damaged or otherwise destroyed if improperly handled [73]. The fragile nature of digital evidence means that sufficient precautions should be taken throughout the investigation in order to correctly preserve the evidence and its integrity. Failure to take into account the fragile nature of digital evidence may render it inadmissible [73] and otherwise unusable [4]. The possible evidence sources which may be relevant to an investigation and used within the investigation depend on the type of incident and the incident itself [75], as well as the setting of the investigation such as whether it is corporate, military or civil [10]. Possible evidence sources include cell phones, PDAs [32], digital cameras, laptops, desktops, memory sticks [73], log files, firewalls, routers [2] and even GPS devices [4] — all of which may contain information of a party's places of interest.

While the focus of our dissertation is on digital evidence, and hence Digital Forensics, it is important to note that physical evidence also plays an important role in these investigations [2][10][12]. Digital data is circumstantial at best and it must be combined with confessions, video surveillance and physical evidence in order to overcome uncertainty with regards digital evidence [13] and allow for further reliability in the conclusions drawn from the digital evidence. These digital devices may contain fingerprints and other biological evidence [4] and care should therefore be taken when collecting these devices in an investigation [73]. Physical evidence provides useful clues and information [75] about the particular incident, how it occurred and who is accountable for it. Ahmad [2] discusses how by simply broadening the scope of a crime scene to include the immediate physical environment around the computing system of interest, it can help to improve the context of digital evidence significantly. Each of these potential digital evidence sources may use a variety of media technology, file systems and formats to store information [41], but how does one go about interpreting, collecting, examining and analyzing such evidence?

Before we can answer this question it is important to discuss the complexity problem as discussed by Carrier [11], and the latent nature of digital data as discussed by Cohen [15]. This particular problem highlights the necessity of digital forensic tools [4][54] in a digital investigation, as well as the overall reliance of investigators on them. They discuss how digital data in its most raw format [11] can be seen simply as a bag of bits or collection of bits [15]. An investigator may through a complex, time consuming manual process, interpret and otherwise determine the meaning of such bits; however, it is inefficient and there exists a high potential for error [11]. The large volumes of digital data [68] — referred to as the quantity problem by Carrier [11] — the probability of interpretation errors and the impacts thereof [11] all encourage an investigator to use a better approach. Digital forensic tools (or simply tools as referred to in this dissertation) are the particular hardware and software that help aid an investigator in this regard and due to the complexity problem and quantity problem, become necessities for an investigator [15]. These tools aid in automating the collection and translation of these raw bits into a more easily understandable form for the investigator to identify, examine [11] and to process the evidence [13]. The investigator then

uses this evidence (digital and physical) in order to draw conclusions [10]. The investigator must however take the converging nature of digital space as well as inculpatory evidence, exculpatory evidence and evidence of tampering [10] into account in their reasoning [15] and the conclusions they draw in order to ensure that their reasoning and conclusions are (more) sound.

2.2.1 The Converging Nature of Digital Space

Ahmad [2] describes how forensic investigations typically consist of two main phases, namely an exploratory phase (induction) and an evidence phase (deduction). In the exploratory phase the investigator focuses on determining hypotheses for what has occurred, how they think it occurred, when they think it occurred, why it may have occurred and who they think may have done it. The exploratory phase plays an important role particularly in corporate scenarios after an incident in order to quickly restore systems to a workable state [64]. In the second main phase, namely the evidence phase, the investigator focuses on collecting evidence in order to serve as a form of proof of the hypotheses they created in the exploratory phase [2].

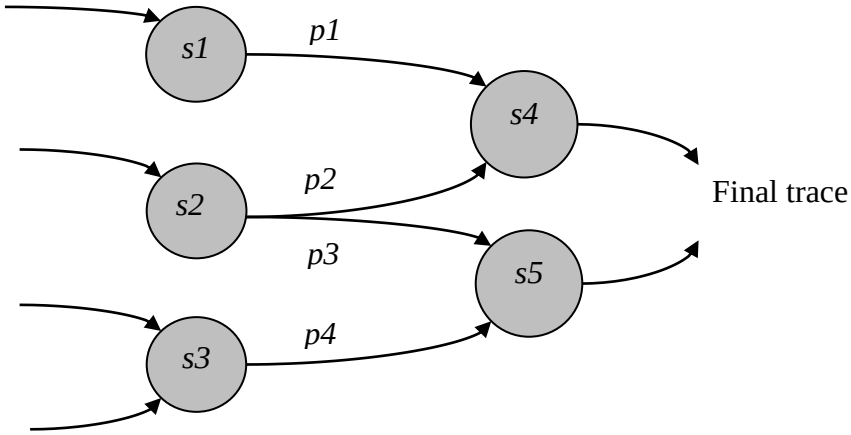
The proof may however not be the same as more formal methods used in scientific approaches, the reasoning being that it may be too difficult due to resource limitations — namely time, processing power, and various funding, particularly as a result of the converging nature of digital space [15]. Before we discuss the converging nature of digital space, we must first discuss the core mechanism behind digital systems that results in this converging behaviour, namely that of a Finite State Machine (FSM).

A Finite State Machine (FSM) uses the current state it is in and the input that it receives to determine what output it will produce and what the next state transition will be [25]. In order to achieve this behaviour and formally describe it, a number of sets are required, namely a set of inputs, a set of outputs and a set of states. The set of inputs describes all inputs that the FSM can receive, the set of outputs describes all possible outputs the FSM can produce and the set of states describes all possible states that can be reached in the FSM. The FSM then makes use of two functions: the first maps an input and a current state to a next state and the second maps an input and current state to a particular output [19]. These two functions allow a FSM to process inputs differently depending on what state they are in. Day [19] goes further to describe how all data communication is a result, or as he states a *side effect* of the execution of these FSMs. Cohen [15] argues in a similar manner and he describes how all digital evidence can be seen as a form of traces — namely *bit sequences produced from the execution of a FSM.*

The execution of a FSM or a digital system can result in different states being entered — based on the next state function and the inputs given [25] — producing a form of partial traces. These various executions and partial traces may converge and result in the same final trace being produced. This convergence makes it difficult to determine the exact sequence of events that produced this final trace [15], such as the bit sequence stored on a digital device. A diagram of this converging nature can be seen in Figure 2. Using a similar argument it may be difficult to determine the exact sequence of events that allowed a FSM to be able to reach

a particular running state of interest, as multiple executions (and state changes) could result in the same state being reached. While it may be possible and feasible to determine possible traces through the FSM that could have resulted in the final trace being produced or the final state being reached over short periods of time, the problem becomes far more complex and difficult as time progresses.

Figure 2: Converging Nature of Digital Space



The execution of a FSM can result in different states *(s1, s2, s3, s4, s5)* being entered — based on the next state function and the inputs given — each producing partial traces (*p1, p2, p3, p4*). These various executions and partial traces may converge and result in the same final trace being produced. This convergence makes it difficult to determine the exact sequence of events that produced this final trace [15].

The converging nature of digital evidence (a subset of all digital data) is one of our main reasons for focusing on capturing the reasoning process of an investigator. By this we mean we would like to capture how an investigator interpreted the evidence, reasoned using the evidence and how the evidence was used to arrive at certain conclusions. Based on the converging nature of digital evidence and the fact that multiple traces could of resulted in the same final trace, we also want to capture how an investigator argues and supports these conclusions, and the various reasoning steps they follow to do so. In the phases of an investigation and in section 2.4 we explore what should be taken into account when forming these arguments and how an investigator goes about forming these arguments, allowing us to learn from them. Our second main reason for capturing the reasoning process of an investigator is the existence of inculpatory evidence, exculpatory evidence and evidence of tampering in an investigation. An investigator must take into account all these types of evidence in order to form sound arguments and conclusions [10].

2.2.2 Inculpatory Evidence, Exculpatory Evidence and Evidence of Tampering

During the evidence phase an investigator may discover three potential types of evidence: inculpatory evidence, exculpatory evidence and evidence of tampering. Inculpatory evidence is evidence that supports the investigator's hypothesis, and exculpatory evidence is evidence that contradicts or otherwise refutes the investigator's hypothesis [2][10]. An investigator must account for both forms of evidence in an objective and thorough manner in an investigation [73] in order to draw more sound conclusions. Evidence that tampering or modification of evidence may have occurred [10] is important, as it emphasizes that an investigator must take care when relying on log files, application files or information stating a particular action was performed by a particular party [75].

An intruder may use a number of means to cover their tracks once they have gained access to a system, including the alteration of files, resetting configurations, disabling of logging, fabrications or complete removals [75]. These form of events are not unique to intrusions, and are concerns for security [52][55] and Digital Forensics [10] in general. Care and consideration should therefore be taken with regards to the interpretation of evidence [73], the reliance on single sources and the reliance on the integrity of files [75]. Tampering and modifications can hence affect the reliability of inculpatory and exculpatory evidence and their existence (particularly if they are fabricated) and is one of the important reasons that we discuss and factor in the certainty of evidence (and an investigator's reasoning), as discussed by Casey [13]. This matter will be discussed in more detail in section 2.4 where we explore the analysis phase further. Before we do that, we will first discuss the various phases of an investigation and the various difficulties an investigator must factor in during an investigation in order to ensure that the integrity of evidence is preserved and that their reasoning process is sound. We discuss each of these phases in detail to provide a thorough background on a Digital Forensics investigation and to begin to emphasize the importance of our research.

2.3 Phases of a Digital Forensics Investigation

Our discussion on the phases of an investigation commences with preparation, and continues to where evidence is returned or disposed of. These phases are preparation, collection, acquisition, examination, presentation, and finally the returning of evidence and evidence disposal. The preparation phase (section 2.3.1) focuses on explaining what is meant by preparation for investigations, the importance of it and the impacts that poor preparation can have on the investigation. The collection phase (section 2.3.2) deals with the various tasks an investigator must perform when collecting evidence from a scene. Our discussion on the collection phase allows us to emphasize how pitfalls so early in an investigation can drastically impact evidence, its admissibility and the investigation as a whole. We then continue to the acquisition phase (2.3.3) which discusses how an investigator creates a particular duplicate of the collected evidence storage media, so that the original evidence source can be properly preserved throughout the investigation, ensuring its integrity. This particular phase allows us to elaborate on the importance of hashing and how it can aid in investigations, as well as the importance of an investigator's tools. Section 2.3.4 then

explains the examination phase, and how it consists of extraction, data reduction and analysis. Section 2.3.4 is important as it allows us to elaborate on how an investigator begins recovering and identifying useful information, and how they use a variety of tools in order to interpret the evidence and begin forming their arguments and conclusions.

The final two phases that we discuss are the presentation phase (section 2.3.5) and the returning and disposal of evidence (2.3.6). Section 2.3.5 elaborates on how an investigator goes about presenting their arguments and the evidence that supports such arguments, to their intended audience. This allows us to learn how an investigator forms such arguments and how the investigator captures and records this information for the intended audience. It also allows us to investigate which information an investigator incorporates when presenting such information and the level of detail that they use. We then conclude our phases of an investigation in section 2.3.6 where we discuss how an investigator returns evidence to its rightful owners and the various aspects an investigator should take into account when disposing of evidence. Section 2.3.6 is however not directly related to our dissertation but it is included for the sake of completeness.

2.3.1 Preparation

The preparation phase is a broad phase and encompasses all the steps which are required in order to ensure that investigators are sufficiently skilled, prepared and capable of performing the various investigations which may arise. Comprehensive training programs play an important role in this phase, and help to ensure that investigators are competent to perform the required tasks in an investigation, and to use the various tools they may require in an investigation [47]. These training programs must ensure that investigators understand the importance of documentation and objectivity throughout the investigation and that they are aware of the fragile nature of digital evidence [73]. Investigators must also understand the impact their actions may have upon such fragile evidence, especially in terms of its admissibility, integrity, credibility [10] and overall spoliation [73].

It is important that these training programs keep up with, among others, the latest developments in technology and storage media as well as new formats, new vulnerabilities, new legislation [75], new manners of storing digital data and new data concealing techniques [4]. They should also keep up with the various malicious techniques currently in use [28] and with developments in forensic tools and data recovery techniques [47]. Ensuring that investigators are well trained in the forensic tools and data recovery techniques that they may require, as well as these other areas listed, helps to ensure that they are better prepared. These tools must be kept up to date [11] and licenses for these tools should be maintained so that they are available when they are required [4]. These tool updates are important for the preparation phase and training in general as they provide fixes for faults and bugs which have been detected [11] through testing and other means.

An important aspect of the preparation phase and of training is a formalized investigative procedure, which explains in detail all the actions an investigator must take during an investigation [75]. Simulating real-world investigations, situations and scenarios helps to provide investigators with real world experience in training and helps to ensure they are

better prepared. It provides investigators with a chance to practice the actions in the procedure, and ensures that they understand and comply with it [75]. Training should also allow investigators to learn from past and existing investigations, as doing so will aid investigators in not making the same mistakes [12] that were made in previous ones, and accelerate the entire investigative procedure [75]. Such lessons should therefore be factored into the formalized investigative procedure. Overall investigations may draw over large periods of time [68] and be costly in terms of resources such as man hours, productivity and funding [24]. Proper preparation can help to accelerate investigations [75] and their effectiveness [64] and hence is an important phase. While preparation is not directly associated with any one investigation, the pre-search phase is.

2.3.1.1 Pre-Search Phase

The pre-search phase is investigation specific and commences as soon as an investigation should take place [73]. It is generally unique for each investigation, particularly due to the fact that each crime scene is unique [4]. The pre-search phase focuses on ensuring that investigator(s) are sufficiently skilled and capable for the particular investigation [73], and for decisions they may be required to make [75] during the investigation. In order to achieve this, sufficient information about the incident and the suspect(s) should be collected. Knowledge of the suspect's skill levels with regards to computers and computing devices is essential to the pre-search phase [75]. Skilled computer users may make use of advanced data hiding techniques especially when covering their tracks. Devices may therefore be found at the scene containing malicious code that, at the stroke of a key, may destroy entire hard drives, delete system logs or other valuable data [32][75]. By informing investigators of the potential skill level of a suspect, it will help ensure that the necessary care and precautions are taken, and that investigators are aware of the potential pitfalls and *traps* that they may encounter.

Information with regards to the types of evidence that should be sought and collected in the investigation, as well as their potential locations provides valuable insight for the investigation. The benefits of such pre-identification of evidence is that it helps to ensure that important evidence is not left behind, possibly resulting in inaccurate [73] and/or incorrect conclusions [10]. It also helps to ensure that the necessary legal authority (search warrant) to search for and seize such suspected evidence, has been obtained [4]. Briefings of this information should be given to all investigators involved in the investigation in order to ensure that they are aware of this information and that they are aware of what potential evidence falls within, and out of, the scope of the investigation. Information about not only the possible locations of evidence but also the environments (physical and network based) that will be encountered also play an important role in preparing for an investigation [73] and in ensuring the safety of the investigators involved.

All the information collected helps to ensure that sufficiently able investigators are chosen for the investigation, and that they are sufficiently equipped with the tools and equipment which may be required for the particular investigation [73]. An inexperienced and/or insufficiently skilled investigator can easily affect the integrity of evidence and its admissibility [4][73] which may affect the outcome of the investigation [13]. Should an organization not be

capable or lack investigators with the required skills and expertise, they can use the acquired information to properly select and recruit the assistance of external consulting agencies to aid in the investigation [73]. The tools are essential as they help to ensure that an investigator can perform the necessary evidence collection and examination in a forensically sound manner. Investigators will also require the use of other forms of equipment in order to correctly perform the next phase of the investigation, namely the collection phase. The equipment includes screw drivers, labels, tape, torches, cameras [73], antistatic bags, cable ties, evidence bags and gloves [4]. Once the pre-search phase has been completed, investigators commence with the collection phase.

2.3.2 Collection

The main goals of the collection phase are to secure the scene, ensure the safety of all individuals at the scene, to document the scene and to identify, seize and preserve all potential evidence — physical and digital [4]. Included in the collection phase is the packaging, transporting and storage of evidence [4][32]. During collection, and in all phases of an investigation, documentation, objectivity and the following of sound forensic actions to preserve the integrity of evidence is required at all times. The collection phase strongly relies on the preparation and pre-search phases, as any actions taken by an investigator can alter the integrity of evidence, and hence affect its admissibility in a court of law [10]. When an investigator secures the scene, they must take control of the area of interest, quickly moving people away from the scene to avoid any evidence spoliation or tampering, and they must allow printers to complete their printing [73]. An investigator must perform all these actions while ensuring the safety of all persons at the scene [4] and the integrity of evidence [73].

During the pre-search phase, investigators will have been prepared for the potential evidence sources they may encounter and how to ensure they collect the relevant evidence in a forensically sound manner. This preparation will aid in accelerating the collection phase of the investigation and will help to ensure that important evidence is not forgotten [75]. The investigator must quickly identify all such evidence (physical and digital), ensuring that sufficient precautions are taken to preserve the integrity of this evidence [4]. This is particularly important for perishable evidence, such as real-time data which may be lost if sufficient precautions are not taken by the investigator [4][73]. Digital devices such as cell phones, keyboards and laptops may also have trace evidence — including biological and latent prints (fingerprints) — and care should be taken to ensure that such valuable information is not lost or damaged [4].

An investigator should search for and collect, provided the search warrant allows, diaries, phone books [73] notebooks, calendars and photographs [4] or other pieces of paper which may contain important information such as passwords and/or dates of interest [32]. Manuals and documentation of digital devices may also play an important role in aiding an investigator to correctly interpret and acquire evidence from the device in a sound manner [73], and should hence also be collected. Certain devices such as PDAs and mobile devices may lose important internal data should their batteries become completely drained [32]. An investigator should therefore retrieve all cables, power leads and cradles of devices to be

seized, in order to ensure that these devices can be sufficiently powered to preserve their valuable information.

During the collection phase an investigator may conduct a number of interviews with the individuals at the scene [73]. The investigator should record any information that these parties may provide, whether in the form of passwords, testimonies and/or alibis [4]. Should the suspect provide any information or advice, it should be recorded as well, and caution should be taken in following any such information, particularly if it relates to digital evidence that was in their possession [73]. The investigator must also document the scene of the incident.

The focus of documenting the scene is to create a permanent, accurate and detailed record of the entire scene [4]. The investigator must document, photograph and possibly video, should such capabilities be available, the entire scene in order to create such a record. The record must capture the layout of the area(s) of interest, the suspect's computing equipment and any other aspects which may be relevant to the investigation [73]. In addition, the investigator should accurately record the make, model, location and conditions of electronic devices — including their orientation, whether status lights were on and whether the devices were warm (demonstrating that they may have been used recently) — of all devices that was identified and is to be collected. An investigator should also document the evidence which will not be collected [32] and the reasons for doing so. It is important to note that not all evidence can be collected, based on the search warrant and for practicality reasons [73]. In certain investigations such as a fraud, simply identifying and properly documenting equipment to perform certain actions at the scene, such as scanners and card skimmers, may aid an investigation by demonstrating that a suspect had the capability to perform these actions [47].

When the investigator seizes the potential evidence it is important that they follow the correct routines and precautions for each particular piece of evidence [32]. By doing so it helps to ensure that the evidence's integrity is preserved as well as its evidentiary value to the investigation. The packaging, transportation and storage of evidence are important steps of the collection phase and the seizing of evidence. These steps focus on the packaging of potential evidence at the scene, transporting collected evidence from the scene to a secure storage facility, and the storage and protection of this evidence [64]. When packaging seized evidence, all evidence should be documented, properly labelled and a Chain of Custody associated with the evidence [32]. The Chain of Custody is an important document in an investigation and is used for capturing all details about the evidence [68]. It plays an important role in keeping track of where evidence is at any particular point in time [73], and Chapter 3 provides a more in-depth discussion in this regard. It is important that the investigator photograph and/or sketch the layout of all computing systems, and properly label their cables and the ports they are connected to so that the system can be reconstructed, if need be, to avoid any criticisms early in the investigation [73]. When an investigator performs these steps (packaging, transportation and storage) it is essential that they take the necessary precautions to ensure that the evidence is protected with regards to its various sensitivities which may affect its integrity [4].

These sensitivities include physical shocks (or bumps), scratches and bending [32], magnetic sources, temperatures (namely excessive heat and cold), water damage, high humidity [4], dust and smoke [73]. Another particular sensitivity is static discharge and it is therefore important that an investigator make use of anti-static bags when collecting evidence. The fragile nature of digital evidence and its various sensitivities require an investigator to perform regular integrity checks on the evidence, so that they can confirm and verify the evidences' integrity and reliability throughout the investigation [4][47]. Certain digitally enhanced devices such as mobile phones and PDAs with cellular and Wi-Fi connectivity and functionality require additional precautions to be taken [73]. These precautions include the use of a shielded box to prevent these devices from connecting to networks [32]. Should these devices (or any other) rely on battery power for internal data, then the investigator must ensure that they are sufficiently charged and handled throughout these steps and the investigation [73].

The potential to affect the integrity of the evidence seized during this phase is very high, should proper care not be taken. It is therefore important that investigators fully document and record all details of actions they took during an investigation to preserve the integrity of evidence. By an investigator doing so, it can help to avoid challenges and scrutiny against the integrity of evidence, as well as their actions [73]. It is for this reason that documentation is an on-going process throughout an investigation as well as why it is so vital [4].

2.3.3 Acquisition

Carrier [10] discusses that the acquisition phase focuses on saving and capturing the state of a digital system in a forensically sound manner so that it can be analysed later. The state of a digital system is composed of two aspects, namely the executing code and the dynamic data in the device's volatile memory (live data) and secondly the data stored on the storage media of the digital device. One can capture the state of the digital device at the scene, or in a secured, controlled location, such as a forensic laboratory. Performing acquisition at the scene is uncontrolled but beneficial as it helps to avoid any damage to the evidence due to battery depletion, storage, transportation and packaging. A controlled environment, however, may be better suited in terms of equipment [32] and facilities to protect against the sensitivities of digital evidence [73], as well as allow for a more detailed, safer and more thorough acquisition.

Live data, namely data that is available while the device(s) are powered on (or in a live state) may contain and provide valuable information for the investigation [73]. Live acquisition focuses on capturing this live data from the volatile memory of the device. The important information that is captured in live acquisition includes connectivity information — such as network connections details and open ports — as well as system and registry information. It includes information about running processes and application, information about users that are logged on, on-screen information, previously entered passwords [73] and temporary files in memory [75]. Information regarding the presence of a Trojan horse or an active back door [75] may also be present in volatile memory, and may play an important role in supporting a suspect's innocence in an incident [13]. In addition, volatile memory may contain already

decrypted data of which its encrypted form may reside in non-volatile memory. Should the device be powered off such information may be lost and may not be able to be recovered.

Once the capturing of all necessary information is completed a binary dump of the volatile memory is performed [73], completing the live acquisition, but the investigator must take care to ensure that sufficient precautions are taken throughout such an acquisition. Sufficient care should therefore be taken as each action the investigator may take may affect the contents of the volatile memory [73]. An investigator must therefore be well trained, skilled and capable of performing such acquisition, to understand the impacts of their actions and the tools they use and to ensure they provide a detailed audit trail, of the actions they take and its effects [4][47][73]. A video recording of the steps an investigator follows may also be used to verify their actions [32] and to provide an additional proof for their audit trail. Once live acquisition is complete an investigator then proceeds with the correct power-off procedure for the device [73], to prevent any (further) modifications to the data on the device's storage media (or medium) [51]. The investigator then creates an exact duplicate of the storage medium (media) of the device.

One creates a copy of the storage medium by creating an image of it using imaging software [32]. Imaging is the process of producing an exact duplicate, namely a bit for bit copy [73], of all data from the source storage medium [47] and it differs from backups in that it operates below the file system level [64]. By creating a bit for bit copy, an investigator can capture all the contents of the storage medium in totality and in its native format — ensuring best evidence [73]. This means that not only are readable contents on the device captured, but so too is the information (and data) stored in free and slack space, deleted data and fragments of documents that may remain on the device, and data that may not be in an identifiable format [41]. Files that are simply deleted from the storage medium, result in their reference point for the file being erased, but not the actual data of the file [41], allowing the file data to be potentially be captured and extracted [10]. **S**hould one capture the data in a non-native format, such as when capturing CCTV evidence, it can result in a loss of quality and important data such as time and date information [73]. Pursuing such an approach results in a copy that is not a bit for bit copy of the source [73], and hence not an exact duplicate [47], affecting its reliability and admissibility in the court of law.

When investigators create such images, it is important that they do so in a secure and safe location [4] and that the investigator takes sufficient precautions to ensure it is captured in a forensically sound manner [51]. The Association of Chief Police Officers [73] states that the same rules that apply to documentary evidence also apply to digital evidence. It is therefore essential for an investigator to ensure and demonstrate that evidence is no more or less than when it was first collected. In order to achieve this, the precautions an investigator must follow include using sufficient write protection (write blockers) to avoid any write requests and alterations to the collected, source evidence [15][39] and the use of strong one-way cryptographic hashes (message digests) [32][64] such as MD5 hashes [51]. These hashes help to verify that the image is indeed a duplicate of the storage media [51] and they provide an efficient means to ensure that the integrity of the evidence is preserved throughout the investigation. This is particularly so as these hashes can be regularly calculated and compared

in order to verify and confirm that no changes occurred to the original or to the copy [4] throughout its lifetime [32].

The image is created and stored, through the use of imaging software, on appropriate target Write-Once-Read-Many (WORM) media [64], which will then be used in the examination phase [47], while the original is kept in storage for safe keeping [15] in order to preserve its integrity [32]. The first copy one makes of the original is called the master forensic copy, and it is used to create additional copies of the image for the examination phase [32]. The master copy is also kept securely in storage and only accessed should more copies be required, allowing the original to remain in storage — helping to avoid any potential scenarios that may compromise its integrity. The original, master forensic copy and the forensic copies of the master copy must all be verified by computing their one-way hashes [32] and confirming that they are equal.

In certain scenarios, imaging the entire device's storage medium (media) may not always be feasible, such as if the size of the storage medium is too large [69][73]. In these particular scenarios one may then consider partial or selective imaging. Partial and selective imaging focuses on the retrieval of a subset of data and information which may be of interest to the investigation. Should an investigator pursue such an approach, it is important that they can verify and argue that all relevant information and data contained on that device was captured. Care should be taken when pursuing such an approach as certain information contained on the medium (media) provides a valuable insight and context for an investigation [2] and should it be lost/not recovered, it may result in misinterpretations and false conclusions [73]. Once the acquisition phase has been completed, an investigator commences with the examination phase.

2.3.4 Examination

The examination phase consists of two main steps, namely extraction and analysis [47] on a forensic copy (image [64]), if possible, of the original evidence data. Extraction focuses on the identification and recovery [4] of data contained on the particular medium/media [73] that is/are collected during the collection phase of an investigation [47] and ensuring that the data is visible to the investigator for analysis [73]. Once the identification and recovery of data is complete it is then processed to extract a subset of potentially useful information [10]. This is known as data reduction and it helps to reduce the amount of data that is extracted to a more feasible amount, so that it can be sent for analysis [73]. Analysis is then performed on the extracted data and focuses on the searching of the data [15], the interpretation of the data and the placement of the data into a logical and useful format [47]. In order to perform these tasks an investigator requires a variety of tools [51]. The investigator then uses these results from these tasks, along with information acquired throughout the investigation, to draw sound conclusions [47]. The National Institute of Justice [47] emphasizes that an investigator must take these results and information in their entirety, rather than in isolation, in order to form a more complete picture of the incident and hence draw more sound conclusions. We will now explore these steps and discuss the importance that they play in our dissertation.

2.3.4.1 Extraction

The National Institute of Justice states that there are two types of extraction, namely physical extraction and logical extraction [47]. Physical extraction refers to the identification and recovery of data on the media at the physical level and hence is independent of the operating system(s), file system(s) and/or application(s) installed or otherwise stored on the particular medium. Physical extraction can be performed through a number of means including keyword searching, extraction of the partition table and unused space of the image of the medium, and file carving [47]. File carving across a physical drive (or associated image) is the process of reassembling [51], recovering and extracting useable data and files that may not be accounted for by the operating system and file system [47] or where the file system is damaged/corrupted or (purposely) removed [51]. The National Institute of Standards discusses how this includes data from unallocated space, data from file slack and files which have been deleted [32]. These particular files and data may contain valuable information for the investigation and hence are desirable sources of information.

Logical extraction differs from physical extraction in that it refers to the recovery and identification of data and logical storage objects (such as files and directories) [32]. The data and logical storage objects are based on the operating system(s), file system(s), and/or application(s) installed on the medium, and hence the data and logical storage objects are dependent on them [47]. The benefits of extracting logical storage objects, is that they are normally easier to understand, interpret and use during analysis [32]. Logical extraction has a wide range of uses including the extraction of protected, encrypted and compressed data as well as the extraction of the directory structure, attributes of files and attributes of the file system stored in the image. It is important to note that the National Institute of Justice discusses how logical extraction can be used to extract and recover deleted data, data from unallocated space and file slack [47]. These capabilities are however discussed during the physical extraction by the National Institute of Standards [32]. The National Institute of Standards goes on to explicitly state that deleted data cannot be recovered with such a logical approach [32]. Regardless of this discrepancy both logical and physical extraction have their benefits and uses and can aid in the extracting and recovering of information for the investigation. The problem, however, is that there may be a large amount of extracted data and an investigator must reduce it to a more feasible amount in order to facilitate a more effective analysis on the data.

The large storage capacities of digital devices (quantity problem) and the fact that these capacities are increasing rapidly [41][69] means that an investigator must be able to, as the Association of Chief Police Officers puts it, *separate the wheat from the chaff* [73]. Once all the data has been recovered through the extraction processes, data reduction must commence. While analyzing all data contained on a device would most certainly be beneficial to an investigation it is not efficient and may not be feasible (depending on the amount of data), particularly due to time and resource constraints [11]. Manson et al. [42] discuss how searching through evidence is time consuming and describe it as looking for a needle in a haystack. They discuss how an investigator should have sufficient tools and means in order to sift through the extracted data and discover important information for the investigation [42].

Data reduction can be performed in a number of ways [11] including the filtering and removing of known files by comparing hashes, sorting by file type, and by using only files which may be of interest and are related to the investigation [47]. For example FTK, a well-known digital forensics toolkit, includes a KFF (Known File Filter), that helps to facilitate the filtering of known files [42]. The Department of Justice [4] provides a list of potential evidence (sources) which may be relevant should certain incidents occur, to help accelerate the search for potentially valuable information. Examples include spreadsheets for fraud investigations, images for child pornography and web activity for online gambling investigations [4]. This helps to reduce the potential evidence an investigator may begin searching for valuable information in the analysis phase.

2.3.4.2 Analysis

Analysis then commences on the recovered data, focusing on the searching and interpreting of the data. The three main types of evidence which must be searched for, discovered and understood during analysis are inculpatory evidence, exculpatory evidence, and evidence of tampering. By an investigator discovering and correctly interpreting these types of evidence it allows them to better understand what has happened, what is meant by the evidence, who is responsible and how it got there [73] — providing the investigator with the necessary means to form more sound arguments. When an investigator searches for evidence they may use string searches, browsing, pattern matching — such as that used in grep [32] — and indexing [64] among others. Indexing is recommended during analysis as it helps the investigator to capture and summarize where important data or related data may be, helping an investigator to quickly and easily locate such information in the future [75]. The investigator then uses various types of analysis, such as timeframe analysis, data hiding analysis and application and file analysis [47], in order to place the data into a logical and useful format [73], to correctly interpret it and to draw conclusions.

An important aspect of analysis is interpretation, as it plays a vital role in the understanding of the evidence discovered by the investigator and the results of tools [15]. A particular concern with interpretation is that there is a potential for errors, a potential for bias, and hence inaccurate conclusions [10][13][73]. In order to help overcome these errors, objectivity must be maintained throughout the investigation [73]. In addition, an investigator's reasoning and conclusions must be based on and supported by evidence [15]. An investigator should also maintain detailed information about the conclusions they make and the arguments behind them [4]. Casey [13] states how investigators should also factor in the certainty that they can place into evidence, and the arguments drawn from them. He discusses how the incorporation of such certainty, periodic integrity checks of evidence and the appropriate documentation help to allow an investigator to be better prepared against possible challenges and scrutiny they may be faced with when presenting their results.

We will now explore examples of analysis that can be performed in order to aid an investigator in the interpretation of the recovered evidence. The examples of analysis cover data hiding analysis, application and file analysis, and time frame analysis. These forms of analysis are not mutually exclusive and must be combined as is needed to achieve optimal results. During these various forms of analysis it begins to become apparent the possible

benefits that may exist when an investigator draws conclusions when treating all evidence and existing arguments uniformly.

2.3.4.2.1 Data Hiding Analysis

Data hiding analysis focuses on the detection, recovery and the analysis of hidden or concealed data on a computer system [32]. A suspect may use a variety of means to conceal data, such as using reserved areas of data storage, such as slack space, and/or areas outside normal file systems and normal user usage to conceal important information [51]. Further methods to conceal data include file extension mismatches, password protection, encryption, compression [32], steganography [47] and redaction [41]. One can use file extension mismatches in order to conceal important data from the investigator, or remove extensions entirely from files. Tools may reveal these mismatches by correlating file headers to file extensions. This correlation can also allow tools to determine the correct file types (during application and file analysis). By the investigator using the correct file types for files, it can help ensure that they correctly retrieve and interpret the content of these files [11], as well as the files' metadata [32].While the decryption of encrypted files may be possible, it may be too difficult and time-consuming given the resources available to the investigator [75]. It is therefore important that as much information as possible be captured during live acquisition since already decrypted files may reside in volatile memory. Another method used to conceal data is redaction.

Redaction is the process of removing sensitive information from a document or set of documents [51], so that it cannot be later extracted or recovered [1], before it is presented to other parties [41]. This sensitive information may include confidential information [51], privileged information, and information maintained with regards to doctor-patient relationships and attorney-client relationships, court proceedings, proprietary information and classified data [41]. Two main approaches can be used to perform redaction, namely a black marker approach and physical removal. The black marker approach involves concealing sensitive information within a particular document. Physical removal refers to physically removing selected documents from a set of documents. Manes et al. [41] go on to discuss how redaction may involve removing entire files and folders (documents) containing sensitive information, and they discuss how this is referred to as black outs.

Redaction may be used by a suspect to conceal incriminating information; however, depending on the approach used not all information may be redacted sufficiently/correctly [1]. The metadata of the redacted file may not be correctly removed and, as a result, useful information may still be contained and recovered in the headers, footers and end notes of the file. Should a suspect attempt to conceal information by changing its colour or by placing graphics over the text, these actions can be reversed and the information easily recovered. An investigator should also examine the comments and the track changes of the redacted document/file (should the file type support and contain such information) to see if any vital information is still present. This type of information can provide valuable insight to how redaction may have been performed and by which party, and may be erroneously forgotten by a suspect. Should the suspect use physical removals, the investigator may be able to recover the deleted data during extraction.

The main requirement with data hiding analysis and analysis in general is that an investigator must have these tools available and they must be well-tested and used correctly. The investigator can then use these tools to detect, recover and transform such concealed information, allowing an investigator to support arguments that the subject had knowledge of, ownership of, possession of and/or intent for these files and their contents to be concealed [47].

2.3.4.2.2 Application and File Analysis

There may be a number of files, applications and programs stored on the collected devices and systems (and hence their associated images), all of which may provide valuable insight for the investigator with regards to the device's/system's capability and the suspect's knowledge of the device/system [47]. These files and programs may help provide information about a suspect's potential uses for the system/device, the suspect's experience using the device and the associated security measures which the suspect may have used. Particular files that may aid in this regard include log files, system files [11] and user configuration files [47]. An investigator may also reverse engineer program executables to reveal further information [11], such as encryption algorithms [52], storage locations and/or attack processes. These files and programs may provide valuable information for the investigation and, as a result, file and application analysis is important for any investigation.

File analysis incorporates a wide range of tasks including the identification of patterns and naming conventions used by a suspect, the comparison of files, the inspection of metadata and the formation of relationships between files. The naming patterns and conventions, and storage locations used by a suspect are beneficial as they may aid in determining potentially relevant files for the investigation [47]. Other forms of patterns can be identified through the use of statistics, whereby tools can be used to determine the number of a particular file type compared to others [47]. These comparisons can be used to possibly determine a suspect's file preferences as well as to show the number of files with misnamed extensions, helping to support the claim that a suspect may have attempted to conceal information. Inspecting the metadata of files can reveal further information, such as authorship [32], the date a file was last edited, whether the file was edited or not, the number of times editing was performed, as well as whether or not a file was printed or not [47]. While these tasks and files may each reveal small pieces of information, they may be used by the investigator to link evidence and more strongly support arguments. One particular relationship which may be of interest when linking evidence is that of a timeline. This leads us to our next form of analysis, namely timeframe analysis.

2.3.4.2.3 Timeframe Analysis

Timeframe analysis focuses on the correlation of files and other evidence in order to create a timeline of when events took place [32]. In order to form such a timeline, particular evidence such as logs (access logs, application logs among others), emails, phone messages, billing information and CCTV footage may be used [2]. The investigator must however take into account that the devices and systems used to record these forms of information may have discrepancies, making an exact timeline of events difficult to create. The investigator should also take into account the delay between when an event occurs and when it is logged [3].

Further aspects which may affect the capability to form such an accurate timeline include the manner in which timing information is recorded, such as whether time zones are included or not and whether heterogeneous formats are used for these time recordings [15][67]. This brief discussion on timeframe analysis emphasizes the potential benefits of combining multiple pieces and types of evidence, and how an investigator should be able to treat them uniformly they can more easily formulate their arguments and reasoning. We elaborate more on analysis, the linking of evidence and the forming of sound arguments in section 2.4**.**

2.3.5 Presentation phase

The presentation phase focuses on the preparation and creation of a detailed record and summary of all the steps that an investigator took in the investigation, the conclusions (or findings [37]) they reached and the manner, with regard to evidence [10], in which they reached these conclusions [32]. In addition, this particular phase focuses on the presentation of this information to the intended audience through the use of expert testimony, forensic reports and depositions [15]. An investigator must include all detailed documentation, notes, photographs, and tool-generated content that is relevant to the investigation [32] when presenting their results, as well as all testing information that verifies the correctness of the tools they used [24]. The investigator must also include all information, such as the Chain of Custody, audit trails and the integrity checks performed [4]. This information helps to demonstrate that the evidence is no more, and no less than when it was first collected and hence can be relied upon [73]. While the other phases of the investigation are strongly based in the technical domain, presentation is based on policy and law [10] and can be seen more as an art than a science [15]. The presentation phase is essential to any investigation and allows us to discuss the important aspects that an investigator must take into account when they present their results and their reasoning.

An important goal for this phase is clarity and it refers to an investigator ensuring that all information they present is sufficiently described so that it can be easily and correctly understood by its intended audience [32]. The particular setting of the investigation will strongly influence who the intended audience may be, as well as how one should present their findings [10]. For example, the audience of a legal proceeding is generally a judge and a jury — whereas internal incidents may include the technical division that is interested in the vulnerabilities and weak points exposed in the incident [13]. Capturing the evidence and the actions an investigator followed using a sufficient granularity and an appropriate approach (hence an art [15]) allows the investigator to present this information in such a manner that is correctly understood by its intended audience. In order to ensure that these presentations and reports are reasonably sized, are concise and easily followed by the audience, the investigator must include only information that is relevant to the investigation [32]. In addition, investigators must ensure that objectivity is clearly conveyed throughout their reports and presentations, as well as through all the phases of an investigation [73].

In our literature survey, the National Institute of Justice [47] discusses the examiner’s report and Rowlingson [58] discusses the case file. We use the examiner’s report and the case file to further elaborate on information that should be included by an investigator during the

presentation phase, and to describe how a report should be structured when presenting one's findings. By doing so, we elaborate on what information should be captured, recorded and presented by an investigator when presenting their findings. The examiner's report is a particular document that contains all vital information about the investigation that is presented to the intended audience. It includes information with regards to the particular investigator(s) involved, all sources of potential evidence, the identity and signature of the investigator (for non-repudiation purposes), the steps taken by the investigator and the results and/or conclusions derived. At the beginning of the examiner's report, a summary of an investigator's findings are stated, providing an overview of them for the reader. In the later sections of the report, more information about the findings is given, allowing the investigator to go into more detail about them and describe how they were discovered and derived. When the investigator describes how these findings were derived and discovered they must provide information about the processes and analysis they performed that played an important role in these findings. The National Institute of Justice goes on to discuss how the report must contain all supporting materials for their findings, such as audit trails, testing documentation, the Chains of Custody and the evidence itself. The inclusion of these supporting documents helps to demonstrate that the findings can be relied upon and the integrity of evidence was well preserved throughout the investigation. The National Institute of Justice discusses how an investigator should also include a glossary into their report and how it helps to ensure that there is no confusion with terms used within the report.

In the case file discussed by Rowlingson [58], he emphasizes how an investigator must include all information about their hypotheses, their logical arguments for or against such hypotheses and the supporting evidence for such arguments. Rowlingson discusses how, in a case file, an investigator must include detailed information of the incident, what happened and the (potential) damages and impacts of the incident. He discusses how an investigator must provide answers to the important questions asked during analysis, and the investigation, namely the who, what, why, when and how an incident took place. An investigator should provide well-formed, and logical arguments for why their answers and hypotheses hold and are credible. These arguments must capture and clearly show the supporting evidence and they must be objective. Such objectivity can be better achieved by an investigator taking into account all inculpatory evidence, exculpatory evidence [58] and evidence of tampering [10] in entirety, and using it in their reasoning and arguments.

Overall, the presentation phase focuses on the importance of an investigator's hypotheses, arguments and the reasoning behind them. It encourages an investigator to record a deeper knowledge behind their conclusions and reasoning, and it helps to provide a strong motivation for the model discussed in this dissertation that helps to facilitate the capturing of such information.

2.3.6 Returning of Evidence and Evidence Disposition

The final phase of an investigation focuses on returning evidence, namely the digital and physical property, as well as the associated information to its rightful owner [32] and/or the destruction of (the remaining) evidence so that it cannot be later recovered [15]. The evidence

and information used in an investigation — depending on the setting and the case — may include important trade secrets, confidential information of certain parties, client related information [15], privileged information [75], military or government information [10] and other sensitive contents. The number of advancements that are/have been made in data recovery and the potential number of tools available means that sanitization of sensitive and otherwise evidential information is an incredibly important part of a forensic investigation and must be done correctly [48].

Sanitization of digital devices and associated storage media can be performed by using tools such as scrubbing tools and file shredding software, which are designed to destroy data [75]. They provide this destructive functionality by overwriting the clusters of the storage medium containing the evidence a number of times. In this way, data recovery is made more difficult, but even after such scrubbing and shredding, the data or at least part of it may still be recovered. In order to preserve the confidentiality of evidentiary information, an investigator may physically destroy the collected devices and storage containing the evidentiary information [29]. This draws the end to our discussion on the phases of an investigation, the importance of an investigator's actions and the tools they use, the investigator's audit trail and documentation, and an investigator's reasoning throughout the investigation.

In the next section we expand on the analysis phase and how an investigator uses various pieces of evidence in order to form their arguments and derive more sound conclusions. One particular approach that aims to achieve this is the Forensic Integration Architecture (FIA) discussed by Raghavan et al. [54] and it allows us to provide an introduction for the next section.

The FIA provides a means for investigators to integrate evidence and its associated information uniformly, independent of source and storage formats [54]. In this manner an investigator can integrate evidence and their associated information from the large number of disparate sources which may be involved in an incident, and hence the investigation. In order to allow the FIA to keep up to date with advancements in digital devices and media, the FIA supports the registration of interfaces and interpreters for the acquired sources. The FIA incorporates the means to allow for the development of assertions and theories and the means to determine the validity of such assertions and theories through correlation with the evidence and real world events and data. The manner in which these assertions and theories are specified is however not discussed. In addition, the FIA does not factor in certainty and it does not capture the reasoning behind why the investigator made these assertions and theories. Furthermore, the FIA does not seem to keep track of existing assertions and theories which are found to hold (such as to use them to confirm further theories and assertions). In the next section we elaborate on expert systems, a fairly well-known approach for handling these issues, capturing facts, and using them to form conclusions with a particular certainty. Expert systems also provide an explanation system which explains how these systems came to the conclusions they did using the facts.

While expert systems appear to be outdated technologies, Luger [38] discusses how they offer a number of benefits to the Artificial Intelligence field and future technologies. While

our dissertation is not based in the Artificial Intelligence field, these expert systems offer a number of lessons for capturing an investigator's reasoning process. Before we delve into such a discussion we will first discuss what an expert system is and how it achieves its purpose. We will then discuss how and why the expert system provides valuable insight for our dissertation.

In this dissertation we use expert systems and their various components, including a fact database, knowledge base (a set of rules), and an explanation system, as a basis for a concrete metaphor for an investigator's reasoning process. We use such a metaphor in order to integrate various literature on how an investigator should use evidence to form arguments, support hypotheses and otherwise derive conclusions. The metaphor also allows us to integrate various literature on how an investigator should factor in certainty into their results and how an investigator should document the entire process.

2.4 Analysis: Expert Systems, Certainty and an Investigator's Reasoning

Expert systems are designed to capture the knowledge of domain experts, so that this knowledge can be emulated [35] and applied to solving problems and complex decision making, usually done by experts, in such a domain [38]. The expert system uses the knowledge that it is coded into it [35] along with a set of facts that it knows about its environment [18] in order to provide recommendations, diagnoses, draw conclusions and/or to determine which action(s) to take at a particular point in time [35]. Expert systems are composed of several different components that interact in order to simulate such behaviour [35]. The components include a fact database, a knowledge base, an inference engine and an explanation facility [18].

The fact database contains all the facts that are known at any particular point in time [35] about the expert system's environment and it plays an important role in the knowledge base [18]. The knowledge base (or rule database) represents the knowledge of the expert system. It is the particular set of rules which are set up by a knowledge engineer in order to capture and represent the knowledge and expertise of a domain expert [18]. These rules take the form of classical IF-THEN constructs [35], where antecedents are on the left side of the construct and consequents on the right. These antecedents represent the conditions which must hold for a rule to be *fired* or otherwise executed and are combined using the logical AND and OR operators [18]. The consequents are placed on the right side of the IF-THEN construct and are used to specify the conclusions that should be made and/or the actions which should be taken, should the particular rule hold (and hence its conditions be met) [35]. Each rule may have a number of consequents, allowing one to specify more than one conclusion and/or action to be taken [18]. For simplicity's sake, from this point onwards, we will be focusing on consequents being conclusions only, rather than actions as well. These consequents are then *fired* by the inference engine and added to the fact database, allowing the system to deduce further conclusions about its environment using these consequents.

The inference engine is the particular component that is responsible for using the knowledge base, namely the set of rules, and the fact database in order to deduce and determine which conclusions to make [35]. Coppin [18] explains how the inference engine may make use of forward chaining, backward chaining or a combination of these approaches in order to perform these deductions. Forward chaining uses the fact database and the knowledge base in order to deduce which consequents it can derive or should take. Backward chaining is used to determine whether it is possible to draw a particular conclusion given a set of facts and the production rules. The system then works backwards from the conclusion in order to determine if such a conclusion can be reached, using a logical path backwards through the rules to a set of antecedents in the fact database.

Forward chaining and backward chaining are of interest to this dissertation, since an investigator can pursue either approach or a combination thereof in an investigation. A forward chaining approach can be used by the investigator where they begin from the evidence, and then use this evidence to draw conclusions [15]. This can be seen as following a data-driven approach or more an evidence-driven approach. Backward chaining can be used when an investigator has a particular hypothesis with regard to the suspect, the incident or a particular aspect of the investigation, similar to the exploratory phase discussed by Ahmad [2]. The investigator then moves backwards, so to speak, using a form of a top-down approach to determine whether evidence can be found to support such a hypothesis (the evidence phase). This approach can be seen as a goal-driven approach [18], since an investigator can decompose the main goal they would like to reach into smaller, more easily achievable goals. Carrier and Gladyshev both make use of such an approach using Finite State Machines (FSMs) [15]. Let us now explore these approaches and see what we can learn from them.

Gladyshev and Carrier both make use of FSMs in order to reason about evidence and the events that occurred during an incident [15]. They both assume that the system, program or particular scenario that is of interest can be represented using a FSM. Gladyshev uses finite state machines to formally reconstruct the sequences of events of interest that are associated with a digital forensic investigation. This allows one to examine and reason about whether the evidence and observations made during an investigation are possible, based on the sequence of transitions of the FSM. One can then evaluate and reason about possible means for achieving how the events occurred by back tracking (goal-driven approach) through the transitions that led to the final state of interest in the investigation, using cause-effect reasoning to confirm and eliminate sequences of transitions that disagree with available evidence. Care must be taken when discarding these sequences and an investigator must ensure that they take into account evidence of tampering when doing so.

Carrier makes use of an extended FSM model that adds removable devices and more complex states and events to identify and represent machine histories [15]. The extended FSM model allows one to describe previous states and events of the computer at a primitive and an abstract level. This abstraction helps avoid the unnecessary complexity of FSMs and makes information more clear. Abstraction is also used in the hyper-events discussed by Kwan et al. [72], and it helps to emphasize how higher levels of granularity can help in simplifying

complex events. Carrier's extended FSM can then be reduced to a FSM so that it is consistent with other works, such as Gladyshev's [15]. Carrier hence emphasizes the importance of backward compatibility and demonstrates how one should build upon existing research. Gladyshev and Carrier however do not take into account the limited resources of an investigation, and ignore time constraints and computational complexity, hence limiting their practical applicability.

An investigator may also make use of a combination of data-driven and goal-driven approaches in order to form their reasoning, such as through the use of a proof by contradiction or the *reductio ad absurdum* rule, similar to how they are applied to logic and Artificial intelligence [18][25]. While an investigator is by no means limited to only these approaches, our discussion helps to demonstrate that an investigator may use a variety of approaches in order to formulate their arguments, to derive their conclusions and to support their reasoning, and we must factor this flexibility into our work. A particular aspect highlighted by expert systems is how facts, whether using a goal-driven and/or a data-driven approach, can be treated uniformly.

Expert systems allow all facts about their environments, and conclusions they draw — which are then added to the fact database and treated as facts — to be used uniformly to deduce (further) conclusions. An investigator uses evidence, whether physical or digital, as building blocks for their arguments, and may use existing arguments and conclusions as building blocks as well. Treating all building blocks uniformly can help simplify an investigator's reasoning process, and can simplify the manner in which their arguments can be formed and captured without them having to worry about syntactic and semantic differences. In Chapter 4 we explore Containers and how they can aid in ensuring such uniformity in an investigator's reasoning process. We now examine the explanation facility of an expert system and how detailed explanations apply to a Digital Forensics investigation.

2.4.1 The Explanation Facility and the Importance of Detailed Documentation

The explanation facility of an expert system provides an explanation to the end user for why a particular rule was fired (or executed), and how the expert system arrived at particular consequent(s) [35]. Expert systems suffer from a number of problems [38], such as their lack of robustness, their lack of flexibility, and that they are domain specific, but one particular problem that is of interest and impacts the explanation facility is their lack of deep knowledge. Their lack of deep knowledge is one of the main reasons why expert systems are unable to infer conclusions/actions without explicit rules and this affects the explanations given by the system. Expert system explanations are generally limited to the steps that the system took, namely the rules it followed in order to reach consequents. This lack of deep knowledge means that the system is unable to provide the end user with actual reasoning behind why such conclusions, diagnoses and/or actions were made. The National Institute of Standards [32] discusses how digital evidence, the tools used, the steps followed, the techniques used, the methodology followed and the reasoning of the investigator in the investigation can be challenged in a court of law and in other formal proceedings. An

investigator must therefore be able to fully capture and explain their reasoning process to the intended audience [73] and their *deeper knowledge* behind it. Detailed documentation provides such a fundamental means [15][47][58].

Documentation is an ongoing process in the investigation [47], beginning as early as the pre-search phase [73]. The Proposed Standards for the Exchange of Digital Evidence [32] emphasizes that all activity regarding evidence should be captured in such documentation. The Association of Chief Police Officers [73] further states that a detailed audit trail of all information regarding processes, tools as well as the evidence involved in an investigation must be preserved and sufficiently protected. Information should be maintained in such detail that a third party will able to repeat the processes followed by the investigator, and arrive at the same result(s) that are presented. In order to do so, all relevant and important information, such as the tools used and their versions, must be captured using sufficient granularity and detail.

Some tools incorporate the means to log an investigator's actions while the investigator uses the tool, creating a chain of handling [42]. Examples of such tools include FTK, and SleuthKit and Autopsy. Manson et al. [42] discuss how Autopsy even allows multiple investigators to be able to work on a case simultaneously, and how it creates separate chains of handling for each of the parties involved. A particular concern however, is if multiple parties work on the same evidence at the same time, and some form of modification or damage occurs to the evidence. Another possibly problematic situation is, if a person would like to easily determine how evidence was handled by these various parties, the person will be required to merge and correlate multiple logs, which requires additional effort. These chains of handlings have a number of benefits and aid in capturing a detailed audit trail of the investigator's actions. However, some tools may store such action information and information about the case in proprietary formats, making interoperability a challenge for the investigator [32]. One final concern that we discuss with regards to capturing a chain of handling is that it must capture an investigator's behaviour accurately and correctly.

Auditing a user's behaviour is complex since even though a user may execute a particular application/process, the application/process may perform actions which may not be according to the user's intention [3]. These actions may be as a result of a user process, or some additional actions associated with the process/application — making it difficult to tell if the actions are due to the prearranged instruction of the application and/or as a result of the user's intentions. Such concerns apply to the digital forensic environment as well, where an investigator uses a particular tool to perform an action. Merely logging the tool involved is not sufficient and more detailed logging must be in place [2] to show the reliability of the evidence before and after the tool's execution and while the evidence is in the investigator's possession.

The logging should capture all relevant data completely and it should capture the data of the investigator's actions, and the steps the tool performed on the evidence to prove its reliability, at a sufficient level of granularity. An important rule stated by Tan [64] is that what is not captured is lost. Having sufficient logging mechanisms in place to capture all this information

helps to overcome problems associated with forgetfulness and fatigue, and also helps ensure that an investigator cannot deny their actions or claim that certain steps were performed. Overall, logging plays an important part in showing that an investigator adhered to various forensic processes and did or did not take necessary precautions during the investigation to ensure that their conclusions and arguments are based on correct evidence [47][73].

In order to protect and ensure the integrity of digital evidence and one's chain of handling logs, sufficient means must be in place. Digital evidence forms the foundation for investigators' arguments and their conclusions, and the chain of handling logs helps to capture and describe the reliability of the evidence and hence affect the reliability of the investigators' arguments and conclusions. Any form of discrepancy will drastically impact the foundation and reliability of these conclusions [47][73]. It is therefore important that sufficient means are in place to protect them from any form of modification, unauthorised access, tampering and fabrication [3].

Methods for providing protection and ensuring reliability and integrity include but are not limited to, access control systems [15], encryption, [3] hashing schemes [67], time stamping and digital signatures [64]. The use of digital signatures is beneficial as it protects the information's integrity through hashes, providing a form of non-repudiation [77] that allows an investigator to be accountable [52] for their actions and their conclusions. It also provides a means of knowing whom to query should there be any concerns with regards to an investigator's evidence, or their reasoning process. Timestamps allow for a detailed recording of when events occurred [14], allowing for a more through reconstruction process. It is important to re-emphasize that investigations may continue over long periods of time [13][68][73], and investigators may leave an organisation or a case, but by ensuring that detailed recordings of their steps are captured this may be less problematic, particularly in terms of productivity. There is however another concern and this relates to the lack of deep knowledge that we previously discussed in regard to expert systems.

By simply logging an investigator's steps and actions, we may fall victim to the same lack of deep knowledge presented in the expert system. Turner [68] speaks of capturing the purpose of an investigator who requires access to evidence in the Chain of Custody. By incorporating the purpose for an investigator's actions, it can help capture the rationale behind them and why they were performed in a certain order — providing a means to capture the *deeper knowledge* behind these steps and ensuring these actions can be better understood by the intended audience. The capturing of more detailed information is beneficial, particularly due to the potentially long durations of investigations [68] and the fact that this valuable information can be lost or forgotten, should it not be captured [61]. Overall, the documentation and its capability to allow for more through reconstruction provides reliability, credibility and forensic soundness to the forensic process followed by the investigator, their results and the investigation overall [15]. This is because a third party can use the detailed information to repeat the actions, and verify their results and findings [32]. Detailed audit trails and reconstruction can therefore help avoid challenges [13] and scrutiny against an investigator's actions in an investigation [73]. They also provide a measure of certainty to an investigator's work.

2.4.2 The Certainty of an Investigator's Reasoning

A well-known expert system is MYCIN [18], and while it was developed in 1984 for the medical field to aid doctors in prescribing the correct antimicrobial drugs for blood infection, a particular aspect of interest is its use of certainty factors [35]. MYCIN makes use of certainty factors to state the certainty with which it makes a certain diagnosis [18]. Casey [13] states that an investigator should take into account the certainty of their evidence and the certainty of conclusions they draw from this evidence. He discusses that by doing so investigators can be better prepared for challenges presented against their work. These challenges may be raised, for example, by attorneys in a legal proceeding, or an opposing party in a formal proceeding. But what factors influence the certainty of an investigator's work?

All digital data, and hence digital evidence, has a degree of error associated with it, particularly in terms of how the data and events were measured and captured. The degree of errors exists due the possibility for time differences, data corruption, data fabrication, missing data, data loss and data destruction [13]. Investigators must take this into account along with potential errors — tool-based and human-based [10] — that can occur during acquisition, extraction, reduction and analysis, when formulating their arguments and conclusions for an investigation. The various tools an investigator may require and use during an investigation may each introduce errors as a result of interpretation and/or tool implementation errors, hence producing inaccurate results which can be damaging to an investigation [10][37]. Each of these potential pitfalls and errors can decrease the certainty that an investigator can place in their arguments and conclusions [13].

Casey [13] explains how, by following a sound approach, an investigator can help reduce this uncertainty. He discusses how an investigator must question their assumptions and create theories which explain the facts. Casey goes on to discuss how an investigator should explore alternate explanations and show why they cannot hold. Carrier [10] explains the importance of testing tools and how such testing can verify their reliability, allowing for a more sound approach. Endicott-Popovsky [24] emphasizes the importance of a particular form of testing, namely calibration testing, and discusses how such testing can aid in ensuring that a tool is correctly configured and functioning correctly. The benefit of testing, and its associated test documentation, is that it helps to improve and establish the certainty that an investigator can place in their results. The documentation also helps to provide proof of these tests, so that the intended audience can clearly be shown the manner in which the investigator tested these tools and that the tools were functioning correctly. In order to verify their results and further ensure their correctness, an investigator should also use a secondary tool (or more) [10][15][42][67][73][76]. The Association of Chief Police Officers [73] and the National Institute of Justice [47] discuss how proper training, experience and understanding of the tools an investigator uses can help to prevent interpretation errors and usage errors, and can help ensure more sound results. An investigator must also use corroborating data from multiple, independent sources [67] in order to support the certainty with which they can make certain claims and their arguments behind them [13]. Let us now explore some literature which further pursues the matter.

Intrusion Detection Systems generally focus on single components or a subset thereof [76]. Sophisticated intrusion attempts may therefore use a series of coordinated events and hence involve a number of components in order to bypass detection. This means that analysis of individual components in isolation may only provide a fragment of the entire attack and the series of events that allowed for the intrusion. The correlation and fusion of such information from multiple components helps the investigator to better understand the attack and to protect the system from such future incidents. Casey [13] describes how, in network environments, a single event or action may result in multiple traces being produced in different locations. Casey explains how it is beneficial for the investigator to take these multiple sources into account as, although it may be possible for an intruder to modify and otherwise tamper with a subset of the traces, it is difficult for the intruder to destroy all of them. Yasinsac and Manzano [75] discuss how the correlation of logging and internal events, such as web and email activity, with external events such as phone records and witness testimony, can help to create a timeline for events (timeframe analysis). Investigators can use this information in order to determine the whereabouts of a suspect and/or to verify/refute the alibi of a suspect. By the investigator taking into account all the (potential) evidence sources it can enable an investigator to make more sound conclusions on evidence and provide more strongly supported arguments [11], however this is not always feasible. We now explore some approaches which link various pieces of evidence in order to reason about what happened and/or to aid an investigator in their reasoning. We provide an overview of these approaches and discuss the benefits and drawbacks should we pursue them.

2.4.2.1 Ahmad's Chain of Evidence

Ahmad [2] speaks of a Chain of Evidence Model that can be used, should an insider inflict damage to an intranet environment. It focuses on the linking of audit logs — namely recorded proof that particular events occurred — and through this linking it creates a Chain of Evidence or more a detailed chain of events that occurred and that resulted in the incident occurring. The main requirements to achieve such a Chain of Evidence are that sufficient means must be in place in order to capture the necessary events and logs, and that these logs are captured in sufficient detail. In Ahmad et al's [3] work dealing with audit management technology, they explain how relating and linking of logs helps to allow one to reveal potential steps and sequences of events, providing greater, more complete information and context for the investigator. The context is essential, as discussed by Casey [13], as if it is not correctly understood it may result in incorrect conclusions being made.

The Chain of Evidence emphasizes that, despite certain evidence containing vital information for the investigation, they are merely a piece in a puzzle. By chaining evidence together, the entirety of the incident and what happened begins to become clearer. Unfortunately, while the model captures the linking of logs and helps others to easily see what forms of data may be needed in order to form the necessary links, it has a drawback. The drawback is that the model does not provide a capability for capturing the rationale and reasoning behind why such events are linked, nor the certainty that these events are linked and can be relied upon. This information may therefore be lost.

2.4.2.2 Cohen's Model

In Cohen's [15] theoretical model he emphasizes the importance of testable hypotheses, and the relationships among evidence and events to support and refute these hypotheses. The model makes use of two forms of consistency to do so, namely internal consistency and demonstrational consistency. Internal consistency is the consistency among various pieces of evidence and demonstrational consistency is the consistency between evidence and events. Cohen uses the range $[-1; 1]$ in order to map this consistency, where -1 represents complete inconsistency, 0 represents no revealing consistency, and 1 represents complete consistency. Information for how consistency values are to be allocated is unfortunately not provided nor is detailed information captured for why an investigator believes evidence, and evidence and events are related or not related. Cohen discusses that there may be a potentially large number of relationships between evidence, and between evidence and events in a digital environment. By not capturing the information regarding the relationships and the rationale behind the consistency values, this valuable information may be lost or forgotten.

Cohen's [15] model also takes into account various resources available in an investigation namely time, money, capabilities and expertise. Cohen incorporates resources due to their capability to limit what can be done in an investigation. Cohen however states that his model does little to clarify direction for an investigator in the investigation. Given that investigations are limited by these resources, direction is essential in an investigation allowing investigators to perform more sound investigations and reasoning, more efficiently and cost-effectively. In Cohen's discussion of the model he goes on to state that legal matters may have outcomes which are inconsistent with adequate logical confirmations or refutations of hypotheses. He therefore goes on to emphasize the need for multiple confirmations and refutations for hypotheses, in order to help circumvent such outcomes, should resources be available to do so — furthering the need for direction in an investigation. An investigator should therefore be able to specify multiple arguments for why his hypotheses are supported (or refuted) when he captures their reasoning process and guidelines should be provided to help aid the investigator in achieving their goals more efficiently.

2.4.2.3 The Evidence Graph by Wang and Daniels

Wang and Daniels [72] propose a graph-based approach, namely an evidence graph, to facilitate evidence presentation, manipulation and automated reasoning of intrusion evidence in network forensic analysis. In order to achieve their goal they make use of preprocessing, the construction of an evidence graph and finally a hierarchical approach for reasoning. The preprocessing makes use of flexible approach to aggregate raw alerts into hyper alerts, helping to reduce the large volume of and redundancy in log files and intrusion alerts in general, helping to speed up analysis. These hyper alerts maintain a one-to-many relationship with the raw alerts so that an investigator can backtrack and examine alerts at a finer scale should they need to. Their emphasis on a flexible approach encourages one to design systems that can grow and cater for future needs, and their use of aggregation allows an investigator to view details at the required level of granularity.

An evidence graph is then constructed where nodes are used to represent hosts in the network that are of interest to the investigation. Edges are used to represent the observed intrusion evidence [72]. These edges are then assigned weights along with other attributes, which is determined based on expert knowledge and used to represent the impact of evidence. They then make use of a hierarchical reasoning framework to perform local reasoning and global reasoning to automate intrusion evidence analysis. Local reasoning is used to infer the roles of suspicious hosts from local observations and global reasoning is used to identify strongly correlated hosts in the attack to derive their relationships. While the evidence graph emphasizes the relationships between evidence and its potential to piece events together, the large number of attacks and their increasing nature make it difficult to ensure that expert knowledge will always be available. Should expert knowledge not be available then it may be difficult to accurately choose weights to represent the impacts of evidence, affecting the reasoning capabilities of the model. While expert knowledge should most certainly be catered for if it is available, one must take into account that it may not always be so.

2.4.2.4 Bayesian Networks and Reasoning

Heckerman [30] states that Bayesian networks have become popular for encoding uncertain expert knowledge in expert systems, but they also offer a number of benefits for the Digital Forensics. Lee et al. [37] state that Bayesian Networks are beneficial in Digital Forensics as they are able to describe the cause-effect relationship among various pieces of evidence and they are able to reason about the uncertainty of digital evidence due to their use of Bayesian probability and Bayesian inference rules. A Bayesian probability refers to a person's degree of belief [30], and when applied to Digital Forensics can represent an investigator's certainty in evidence and relationships between the evidence.

A Bayesian network is a causal directed acyclic graph consisting of nodes with directed edges (links) between them [37]. These links specify a direct cause-effect relationship from parent to child node [30]. Each of these nodes represents a random variable, which may have alternative states (or values). Lee et al. [37] for example use the states true, false and unknown for proposition variables. Child nodes are then assigned conditional probabilities, which represent the probability that the child node will be in each of the possible states, for each combination of states of its parent nodes [5]. Nodes that have no parents are simply assigned probabilities that they will take on their particular states. Kwan et al. use such Bayesian networks in order to link digital evidence to a hypothesis in an investigation [37] to determine how a crime was committed and to yield a probability of guilt [15]. Lee et al. [37] propose a practical methodology for transforming the findings in forensic reports to a graphical representation using a Bayesian network.

De Vel et al. [20][37] make use of a Bayesian Network combined with a Hidden Markov Model in order to track and predict the degree of criminal activity as it evolves over time. They describe how, in an investigation, forensic entities such as word documents and email messages may have complex relationships with a number of other forensic entities. Each of these entities may be rich in content and meta-data (attributes and timestamps) which provide valuable insight into how these forensic entities are related, and are useful in providing the link between a crime and its victim and a crime and its perpetrator. De Vel et al. then use this

information as input for their model. Their incorporation of metadata of the various forensic entities emphasizes the importance that such information can play in determining relationships and in an investigator's reasoning.

In Lee et al's [37] paper on the use of Bayesian networks to transform the findings in a forensic report to a Bayesian network they discuss a number of pitfalls. Many of these pitfalls apply to Bayesian networks and Bayesian probabilities in general. They discuss how pitfalls can occur if evidence and causal links are missed, if causal relationships are incorrectly placed, if cyclic dependencies exist among the nodes and if evidence is duplicated and reported more than once. They go on to discuss the various difficulties with objectively assigning conditional probability values, particularly when there are many factors to take into account, and they also raise concerns about fairness. Lee et al. also discuss the impacts that the number of conditional probabilities in large and complex Bayesian networks have on computational time [37]. Heckerman [30] goes on to state the concern about the precision of Bayesian probabilities, and asks what the significant difference between 0.601 and 0.599 is, emphasizing his concern. Heckerman's discussion helps to emphasize the need for capturing semantic information regarding weights and why they were chosen.

From all of these discussions we have learnt that pieces of evidence may have complex relationships with one another, and it is through these relationships that an investigator can begin to piece together and reason about the various events that led up to the particular incident. We learnt that there are a number of ways to cater for an investigator's reasoning, each with its own set of benefits and drawbacks, and our investigation of these approaches allows us to learn from them. We also learnt the importance of metadata and its capability to reveal vital information and relationships. In the next section we discuss conflicts in expert systems and how conflicts can occur in Digital Forensics investigations and in an investigator's reasoning. We then explore how such conflicts can be handled.

2.4.3 Conflict Resolution in Reasoning

An important aspect of the expert system is the control strategy which plays a role in conflict resolution [35], and is achieved through the use of meta rules [18]. These meta rules have a higher priority over normal rules and describe how conflicts may be resolved. Conflicts occur when more than one rule may fire, and while in certain scenarios such behaviour can be beneficial, in others — such as when prescribing medication — it is not and these conflicts must be resolved [18]. The possible means for doing so include the assigning of priority levels, firing the rule with the longest matching strategy (in terms of antecedents) or favouring the most updated rule over others. The most appropriate strategy is dependent on the particular conflict which may occur and the particular expert domain.

The potential conflicts an investigator may encounter in their reasoning and arguments are a result of inculpatory evidence, exculpatory evidence, and evidence of tampering, among others. Each of these pieces of evidence may carry a different certainty and weight in the investigator's reasoning and they will play an important role in an investigator's arguments and the conclusions they draw from them. We must therefore not only allow an investigator to capture the certainty of evidence and the conclusions they draw, but also how they

weighed such conflicts against each other in their reasoning. Overall it is important that we attempt to capture *deeper knowledge* behind an investigator's reasoning, so that it can be better understood and more easily replicated by a third party.

2.5 Chapter Summary

In this chapter we discussed what is meant by Digital Forensics and digital evidence, and how a Digital Forensic investigation may incorporate physical and digital evidence. The incorporation of both physical and digital evidence allows an investigator to formulate a more complete understanding of what has happened and to formulate a more sound reasoning. An investigator must however take into account the converging nature of digital space into their reasoning and inculpatory, exculpatory, and evidence of tampering. The converging nature demonstrates the difficulties of forming an exact proof of what happened, and encourages an investigator to use multiple sources to support their claims. The use of multiple sources is further encouraged by the potential for inculpatory evidence, exculpatory evidence and evidence of tampering which an investigator may encounter during their investigation, and which should be taken into account in order to avoid inaccurate conclusions. Taking into account multiple sources and using them to form more sound arguments allows an investigator to have a higher certainty in the conclusions which they draw.

Our exploration of the phases of an investigation has allowed us to discuss how an investigator's actions can drastically affect evidence, their reasoning and the investigation in general**.** Each of the investigator's tools and their actions may drastically impact evidence, its integrity and hence its admissibility. An investigator is therefore accountable for each of their actions during an investigation. An investigator's arguments and conclusions play an important role in implicating and exonerating a party, and potentially negatively influencing their life. This helps to provide a background for Chapter 3 where we explore a means to ensure that an investigator is directly associated with each of their actions and that they cannot easily deny their involvement (non-repudiation) in these actions. We refer to this means as a Proof of Action and it helps to provide our first step towards capturing an investigator's reasoning process.

Throughout this chapter the number of tools, and their associated transformations, that an investigator may require were elaborated on, as well as the potential impacts they may have on evidence and an investigator's interpretation and, hence, the arguments and conclusions they derive. Our various discussions on the importance of capturing the tools used, the transformations performed, the manner in which the tools were tested and overall the reliability of the tools used, helps to establish the need and background behind Chapter 4:The Transformation System. Overall this chapter provides a background for the dissertation that provides a context for the upcoming chapters and elaborates on how they fit together.

Chapter 3

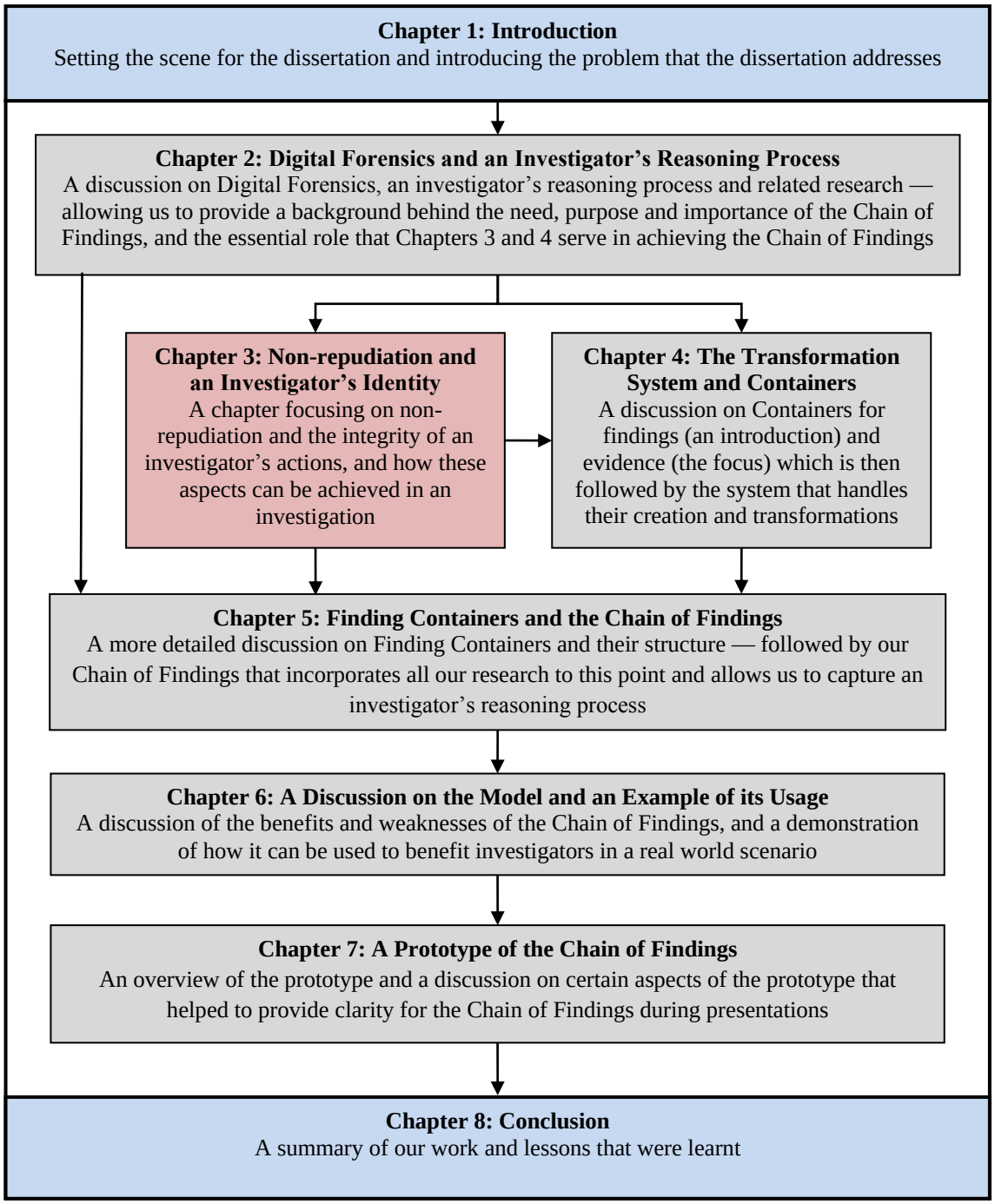
Non-repudiation and an Investigator's ID

Objectives

- To provide a means to accurately record an investigator's actions
- To ensure the integrity of an investigators' actions are preserved
- To ensure non-repudiation of an investigators' actions
- To provide a first step to capturing an investigator's reasoning process

Perfection is achieved, not when there is nothing more to add, but when there is nothing left to take away
Antoine de Saint-Exupéry

A Map of our Current Location in the Dissertation



Chapter 3: Non-Repudiation and an Investigator's ID

3.1 Introduction

In communication systems, the use of a proof of origin (also known as evidence of origin [34]) and a proof of receipt (also known as evidence of receipt [34]) offer a means for neither the sender nor the receiver to deny their involvement in the communication [14]. (We use the terms proof of receipt and proof of origin instead of the terms evidence of receipt and evidence of origin so that we can avoid any confusion with regards to evidence in the forensic sense). These proofs use digital signatures to place a seal over the message and its contents, and then undeniably and unforgeably associate the participants' identities to the message and its contents [53]. These proofs also use timestamps so that one can determine when the communications took place [14] and to help prevent these proofs from being reused [52].

In digital forensic investigations, the Chain of Custody is an important document that is attached to each piece of evidence [68]. It is an important means for recording information about evidence in custody, for keeping track of where evidence is at any particular point in time [73] and for providing a proof of custody. In order to keep track of evidence, it is used to record the investigator (and their information) who checks in or checks out evidence, (possibly) the purpose behind doing so [58] and the date and time that this event took place [68]. The Chain of Custody then requires the investigator's handwritten signature to provide a means of non-repudiation for their action [41][68]. Unfortunately, these signatures and the paper based approach are subject to forgeries [52], and these signatures alone do not associate the investigator with the evidence, or the actions they performed, in the same manner as digital signatures do.

Initially, evidence is captured during the acquisition/collection phase as a bag of bits [15], and can be seen as data that will need to be processed to reveal its meaning [57].The digital data is latent by nature [15], and requires the investigator to use a collection of tools to process the data and to correctly determine the meaning of the data [11]. The tools' functionality can vary between categorizing and indexing the data, recovering deleted files [42], organizing the data in a certain manner to reveal *hidden* patterns, or making use of statistical inferences to draw conclusions on the data [57]. An investigator may utilize these tools' output as further evidence [11], and their interpretation of the results can hence affect conclusions they may draw [15]. There is also the fragile nature of evidence to take into account during the investigation, and therefore the possibility for spoliation [73].

The Proposed Standards for the Exchange of Digital Evidence states that an individual is responsible for all actions taken with regards to the digital evidence while it is in their possession [32]. From when evidence is collected, to when it is presented in the court of law, and finally disposed of or returned to its rightful owner [15], evidence may be altered,

damaged or otherwise spoiled [73]. The tools an investigator uses to understand and transform evidence into a more suitable form and the manner in which these tools are used, calibrated and tested can all have an impact on the integrity of the evidence, as well on the results produced by these tools [24].

Evidence serves as building blocks for an investigator's reasoning, and is therefore fundamental for an investigator's reasoning. Sufficient means must hence be in place to protect and ensure the evidence's integrity throughout the investigation helping to avoid challenges and disputes against them, as well as against the foundation of an investigator's arguments. Sufficient means must also be in place to capture and determine which investigator(s) are accountable for evidence and how they acquired it. The tools used, the manner in which they are used and how calibrating and testing are done may also have impacts on evidence, an investigator's results and therefore their reasoning process.

Overall each of the actions an investigator performs, plays a vital role in influencing and forming part of the building blocks they use, the manner in which evidence is interpreted the conclusions they derive, the arguments they form and overall the reasoning process of an investigator. It is therefore important that we provide a means to capture the steps an investigator follows, the actions they take, the building blocks they use, the hypotheses they create, the conclusion they draw and the arguments they formulate due to their importance in the reasoning process. It is also important that we provide a scheme that undeniably and unforgeably associates an investigator with such actions, and that places seals over these actions so that they cannot be easily modified or altered.

Associating an investigator with their actions helps to provide accountability for their actions. Investigations may draw over large periods of time [68] and involve a large number of people [73], and by linking an investigator to their actions, arguments and contributions, it enables future investigators to easily determine who is responsible for them. These new investigators can then more easily query the appropriate party, improving the efficiency of the investigation. The enforcing of such accountability may also encourage investigators to follow a more forensically sound reasoning process. In order to capture the reasoning process followed in an investigation, it is also important that we capture exactly when actions took place during the reasoning process such that an investigator cannot deny when they performed certain actions. This is important since the sequence of actions may have an impact on the reasoning process, such as if a tool is tested and/or calibrated only after the tool is used. Such a sequence can drastically affect the evidence and the tool's results and, hence, impact the investigator's reasoning process. The inclusion of timestamps has the added benefit that it provides us with a means to construct a timeline of when these actions took place in the reasoning process, allowing for a more thorough means for reconstruction.

The inclusion of seals, signatures and timestamps of these actions helps to ensure that all steps recorded in the reasoning process cannot be challenged on their integrity, provided that sufficient precautions and infrastructures are in place, and therefore creates a Proof of Action. This Proof of Action helps to provide a strong form of reliability for the captured reasoning process, aiding in its presentation as expert testimony in the court of law and it is therefore

important for us to pursue. In this chapter we will be discussing how such a Proof of Action can be constructed, allowing us to provide our first step in capturing an investigator's reasoning process.

In order to discuss how we can construct such a proof we will begin with a more detailed discussion on the Chain of Custody in section 3.2. The investigation of the Chain of Custody allows us to go into more detail about the document and its various elements as well as the benefits they offer. Our investigation of the Chain of Custody allows us to learn from the Chain of Custody and incorporate our lessons into our research. Section 3.3 then discusses identities and digital signatures and how they can be used to create proofs of origin and proofs of receipt in non-repudiation protocols. We then discuss how we can use these schemes to construct a Proof of Action. Section 3.4 focuses on discussing timestamps and the benefits of having them generated by a Time Stamping Authority. We then discuss how one can construct such a Proof of Action, based on the work of Coffey and Saidha [14], in section 3.5.

3.2. The Chain of Custody

The Chain of Custody is an important document [58] in a digital forensic investigation and it is associated with each piece of evidence. It is used to capture the provenance [68] of evidence from when it was seized, packaged, transported and stored [32], to when the evidence is presented in the court of law, and finally, when it is returned to its owner or otherwise destroyed [12][15]. This information includes logging of accesses to the evidence [58], how evidence was stored and what means were used to protect the evidence in storage. The means to protect evidence in storage may include access control systems [64] (card systems [32], biometrics [52]), encryption, physical security [58] and cooling systems used [15]. Included in the Chain of Custody are the physical attributes of the media that was collected, such as serial numbers and model numbers. Additional information such as the capacity of the device, block sizes, and format information are also useful and should be recorded [64]. Tan [64] discusses how he includes the exact commands used and the tools used when collecting such evidence in the Chain of Custody to verify its integrity and to avoid disputes regarding evidence collection.

There are many phases in an investigation, so how does one verify the integrity of evidence throughout these phases? The fragile nature of evidence means that spoliation can occur in any of the phases of an investigation and the Chain of Custody helps to counter such a problem, along with relevant documentation [73]. One of the elements that is important to help show this, is the hash of the evidence that is calculated when the evidence is collected and recorded in the Chain of Custody [12][68]. Hashing schemes are one-way functions that place a seal over the hashed contents and produce a Modification Detection Code (MDC) [14] known as a hash or a message digest [52]. The use of such a seal helps to ensure the integrity of the hashed contents, as any modification to the originally hashed contents will result in a different Modification Detection Code being produced, making alterations easier to detect.

One can then use these Modification Detection Codes to verify the integrity of evidence throughout the investigation, from when it is collected until it is disposed or returned to its owner [73]. One can further confirm the integrity of evidence throughout the investigation by performing and logging periodic integrity checks [58] and integrity checks when evidence is accessed, checked into storage and when evidence is checked out of storage [12][32]. The performing and logging of such integrity checks helps to overcome disputes and challenges against evidence, by showing that the integrity of evidence was an important aspect of the investigation and a priority for the investigator. It also helps to show that sufficient integrity checks were performed throughout the investigation. The logging of such information therefore helps to provide a proof of the integrity of the evidence throughout the investigation. The logging of such integrity checks, strict control over evidence storage [32], and the Chain of Custody all help to verify the provenance of evidence, should it be challenged [68]. The Chain of Custody also includes vital information that allows one to keep track of and maintain a detailed record of who had access to the evidence and, possibly, the purpose for requiring such access [58].

These tracking entries include information about the dates and times these accesses, check-ins and check-outs occurred, the names, contact information and ranks of the investigators that were involved with the evidence, and their handwritten signature [68]. The recording of the dates and times of when these events took place helps to provide a form of a timeline for the evidence, allowing one to determine where evidence was at a particular point in time [2]. Investigations may draw over large periods of time and the use of contact information helps to make it easier to contact the appropriate investigator, should any queries need to be addressed [68]. Recording the purpose for the investigator requiring access to the evidence can help aid in this regard as well. The investigator then uses their signature as a means to verify that the information is correctly recorded [52] and to verify their involvement in the access, check-in or check-out [68]. These signatures can therefore be used as a means of non-repudiation [64] and can be seen as a form of acceptance of accountability for the evidence.

Handwritten signatures can provide a form of non-repudiation in the Chain of Custody; however, handwritten signatures and physical documentation can be forged [52]. These forgeries affect the integrity of these documents and signatures [52] and affect their capability to provide non-repudiation [55]. These handwritten signatures also do not directly link the signature of the investigator with their identity, or the evidence they are accessing, checking in or checking out. Digital signatures are the electronic version of these traditional handwritten signatures [52][53]. They provide a means to directly associate the identity of the investigator with the evidence they accessed, checked in and out, and they provide a stronger form of non-repudiation for the investigator's involvement with the evidence, provided that sufficient measures are in place as discussed in section 3.3. These digital signatures may also be combined with timestamps such as those generated by Time Stamping Authorities to incorporate the time that the signing took place, as has been done in the non-repudiation of communicating entities [14].

Non-repudiation in communicating entities, namely between a sender and a receiver refers to the capability to prevent either of these parties from denying their involvement in the

communication [34]. In order to ensure such denial is prevented, one must generally ensure that the sender of the data is provided with a proof of delivery and the recipient receives a proof of receipt [40]. It is important that non-repudiation systems undeniably and unforgeably form these proofs such that they associate the parties' identities, with the particular communication and the exchange of data in the communication [14]. By associating the parties' identities with the particular exchange of data in the communication, it helps ensure that they cannot deny their participation in the exchange. The particular means for achieving such proofs is a digital signature. Digital signatures, being the primitives for non-repudiation, are applicable to a wide range of fields and not only in communication domains [52]. We will now discuss how digital signatures are used in the communication domain, and how we can use similar means to associate investigators with their actions in order to create a Proof of Action.

3.3 Digital Signatures for Information

Digital signatures rely on the use of public key cryptography and Modification Detection Codes [52][53]. In public key cryptography, each entity is given a unique key pair, namely a public key — which is publically distributed — and a private key/secret key, which the entity must keep secret [25]. In order to digitally sign information and confirm that it is from the entity, the entity first appends their identity to the information that they will sign. The inclusion of the entity's identity plays an important role in decrypting a digital signature and in terms of certificates. The digital signature then takes place in two steps. In the first step the Modification Detection Code is calculated for this new information — which is then unique to this information [52]. In the second step, the Modification Detection Code is encrypted using the entity's unique secret key [52], producing a digital signature for the information [14].

There are a variety of hashing schemes one can use in order to produce the Modification Detection Codes, such as MD4, MD5, SHA-1, HAVAL, RIPEMD, RIPEMD-160 [53]. Care must be taken to correctly choose the hashing scheme that best suits one's needs, as each scheme has its own weaknesses and strengths [52] as well as different processing and storage requirements [77]. In the model presented by Schatz and Clark [59], both the Modification Detection Code as well as the hashing scheme that was used to generate it, are recorded. The use of such an approach is beneficial as it does not restrict the hashing schemes that may be used, and it ensures that the hashing scheme that was used to generate the hash is recorded as well (for verification purposes).

The particular signature generated can then only be properly decrypted and interpreted by making use of the entity's public key [52], revealing the Modification Detection Code for the information. The use of the entity's identity in the information allows one to know which entity signed the information, and hence which public key to retrieve. One can then confirm the integrity of digitally signed information by re-computing the hash for the signed information and confirming that the newly generated Modification Detection Code is the same as the original [14]. Digital signatures are thus able to provide a strong means of non-

repudiation and are unforgeable — provided that there is an adequate public key infrastructure in place and that the required security and protection of secret keys is ensured [34] to prevent key exposure [46]. This is due to the fact that only the entity's public key would be able to correctly decrypt the signature, and produce a Modification Detection Code which would match that of the information. This refers to the authenticity criteria that a digital signature must comply with [52]. There exists further criteria that must be adhered to in order for digital signatures to serve their intended role, namely digital signatures should not be forgeable, alterable and reusable.

In order to overcome concerns of false signatures and to verify that a particular public key does indeed belong to an entity, an approach such as certificates may be used [14][52]. These particular certificates unforgeably bind an entity's identity to a particular public key [52]. These certificates are handled and issued by a certificate authority, which must be highly trusted in order for the scheme to work. If an entity's key is exposed, all the benefits and reliability offered through the use of digital signatures are then compromised [46]. Should a key be compromised, the certificate authority will then revoke that public key and any future digital signatures which try to use the public key are then invalid. In this way, certificate authorities can verify that the public key is secure and correct [14]. A certificate also has a particular time period for which it is valid. The use of such time periods is beneficial as, should a private key be compromised, fabrications can only occur during that period. Certificate Authorities must, however, maintain a certificate history to provide verification of non-repudiation of signatures generated in the past that rely on previous expired certificates. We now discuss how one can timestamp these signed messages so that they can provide a consistent and accurate recording of these events.

3.4 Time Stamping Authority

Ahmad [2] discusses how one can link evidence to create a chain that allows one to better describe and determine how an event took place. He emphasizes the importance of time and the recordings thereof, because of the vital role it plays in enabling such a chaining of events. In Ahmad et al's [3] work on logging systems they emphasize the importance of the manner in which time is recorded. They discuss how sufficient means must be included to ensure that timing information is captured accurately and in sufficient detail. The Association of Chief Police Officers discusses how time is an essential means for determining when events took place, such as when access requests took place and when certain software was used [73]. Tan [64] and Casey [13] also describe how failure to take into account the granularity of time recordings, timing accuracy, the inclusion of time zones and sufficient recording methods can result in a great deal of confusion when interpreting the recorded events.

Tan [64] in his 2001 work discusses how the standard convention for recording time in incident response is through the use of Greenwich Mean Time (GMT), as it allows for a recording of time zones and their offsets. Such a means for recording allows the system to cater for situations where multiple investigators may be working in different locations in the world on a particular investigation. It also applies when a single investigator may travel to

multiple locations around the world in order to complete an investigation. The use of Coordinated Universal Time (UTC) has, however, replaced GMT as an official standard time [66]. The benefit of using UTC time is that it is universal and hence independent from time zones. One can use UTC time to calculate local time by simply adding or subtracting one's local time zone — in this way conversion of time to local time is an easy and efficient process [66]. The benefit of using a universal time scheme is that it makes it easier to construct a timeline of when events took place in the reasoning process, even if investigations span a number of global locations.

When capturing an investigator's reasoning, we want to capture in detail the steps they followed so that they can be accurately reconstructed by a third party to confirm or deny the steps, allowing for a more rigorous, scientific means of proof. One of the important elements that we want to capture is the time these events took place, allowing us to create a timeline of when certain actions took place [73]. An investigator may use a number of tools and a number of systems that may not all be synchronized to use the same time, that may use different methods for recording time (and events) and that may not provide a secure means for protecting such time recordings.

The security of these recordings is important as it helps to ensure that they cannot be easily fabricated, that the integrity of the recordings is preserved [13] and that they can be relied upon, and hence are tamperproof [14]. Tan [64] explains how the most obvious manner to implement timestamping in a secure manner and ensure their integrity, is to digitally sign them. Tan [64] explains how one can use a form of a Digital Notary to provide such functionality. The Digital Notary is a trusted third party that is responsible for generating a seal (Modification Detection Code) for the message, appending the correct timestamp and then signing the entire package. In this section we discuss the Time Stamping Authority (TSA), that functions as a Digital Notary and provides a centralized means for timestamping events [14]. Another possible approach is using a centralized method for synchronizing the systems/devices that an investigator may use, such as that used in the Network Time Protocol [64][66].

In the centralized approach for synchronizing devices/systems, one relies on the use of servers to correctly synchronize the connected clients and the clients that the server communicates with [66].Tan [64] discusses how an important concern with such an approach is the accuracy of the time on the servers. He discusses how one can use Global Positioning System (GPS) Receivers that are attached to the servers with appropriate software. These servers will then receive an accurate time signal from satellites through the use of the GPS device, allowing the clients of these servers to be configured with more accurate times. Tan goes onto discuss how the integrity of these time signals is not easily challenged in the court of law. The benefit of using such a GPS scheme is that it is cost effective [64] and can be applied to any time-interested system. The GPS scheme may hence be used in a centralized logging approach as well, allowing the centralised approach to reap the same benefits of accuracy and integrity.

The main concern with such a synchronizing approach is that, should we utilize it, the investigator(s) and the systems they use will be responsible for generating the timestamps. This means that there is no easy means to ensure that fabrication of these timestamps does not occur, especially if the approach used is not secure [13]. The use of multiple systems to perform the signing and timestamping of events means that they may each be configured to use a different approach for recording these events and the granularity thereof [64]. This lack of a uniform approach may lead to confusion and misinterpretation of when events took place. Small discrepancies may also exist between the times used by these systems and, coupled with the lack of a uniform approach, may make it difficult to form a timeline of actions that were performed [2][3]. The centralized approach for timestamping, such as that used in syslog (http://linux.die.net/man/3/syslog), helps to overcome these problems but it is not without its own set of drawbacks.

Casey [13] explains how most syslog servers receive a log entry from a remote system over a network and then they generate a date/timestampfor that log entry. The benefit of using such a centralized means for handling and performing the time recordings is that only a single location is responsible for ensuring that time is correct, helping to avoid synchronization problems [64]. This approach helps to provide a more consistent means for recording time [14] than if the investigator was responsible for it and removes an investigator's role (and the systems they use) from the process. It also ensures that time is recorded in a uniform manner [13][64], allowing the time recordings to be more easily understood and interpreted. Care must be taken when configuring these centralized servers since, if they are incorrectly configured, then so too are their recordings [64]. The use of GPS receivers can help in this regard, and help ensure that these centralized servers use the correct time and that it is accurate. Casey [13] discusses how, through the use of digital signatures, one can ensure such an approach is (more) secure, and how it allows one to place a higher certainty in the integrity of its recording. The Time Stamping Authority uses such a centralized means for generating and appending timestamps.

The Time Stamping Authority stamps signed messages from an entity by appending its particular identity and a timestamp, and then signing the entire package. The Time Stamping Authority does not consider the contents of the received message [34]. By this we mean that that the Time Stamping Authority is not concerned with the validity of the message, its originator or any other aspects of the message, or its contents. The Time Stamping Authority's role is simply to append its time recording to the message. This helps ensure that the Time Stamping Authority is objective during the time stamping and signing process. It also helps to simplify the Time Stamping Authority and the particular tasks that it must perform [14]. The simplification benefits the implementation and processing time required to perform these tasks. This helps the Time Stamping Authority to generate signed timestamps more efficiently that more accurately reflect and capture the time that the Time Stamping Authority received the signed messages in response to stamp requests. The Time Stamping Authority does, however, ensure that an investigator's certificate is valid, and hence that the public key used is acceptable [14].

Investigators and the systems they may be using may have slight differences in time, due to their configuration and/or due to them operating in different locations in the world. It is beneficial to ensure that time is consistently recorded and captured across the entire investigation [64]. The Time Stamping Authority therefore helps to provide a uniform means for capturing time, and allows these aspects to be taken care of for the investigator. The main concern with such an approach is that, should the Time Stamping Authority be down, or out of service, it can result in a number of concerns for the recordings of an investigator's actions and the recordings of an investigator's reasoning process.

The major drawbacks with using the Time Stamping Authority is that it provides a single point of failure [52] and a bottleneck [34] for timestamping due to its inline role in each of the transactions. One can overcome these problems by using multiple Time Stamping Authorities, each correctly synchronized using the appropriate means, such as a secure NTP or GPS receivers. By attaching GPS receivers to each of these Time Stamping Authorities we can ensure that they each use accurate times. The Time Stamping Authority can then be expanded as is needed to cater for demands while still ensuring that the time stamping is not handled by the investigator, avoiding such issues and complications. By running the same software on all of these Time Stamping Authorities, one can also ensure that they all record time in the same manner. The use of such an approach helps to overcome the bottleneck and the single point of failure — and also ensures that, regardless of which Time Stamping Authority stamps a message, it will have a consistent and correct time recording. These Time Stamping Authorities also append their IDs when they append their timestamps and sign the package, so that it is easy to determine which Authority was accountable for a particular timestamp. One can also use such a means to verify that signing is done by reliable Time Stamping Authorities. Overall the exact method that should be used will depend upon the environment in which it will be used and will require a deep investigation of the benefits and drawbacks to determine which approach is most suitable.

Our focus is to show why such an approach was chosen. For the sake of simplicity, we will assume there is a single Time Stamping Authority and will leave these concerns for further research. Before we can discuss how the Time Stamping Authority will aid in the creation of the Proof of Action, it is important to note that there is a slight delay between when the message is sent, and when the timestampis applied [13]. These delays are mainly as a result of travelling time and processing delays [66]. It is hence important that sufficient mechanisms are in place to ensure that these delays do not affect the accuracy of the recording and that sufficient means are in place to reduce such delays. It is also important to note that the timestamp will convey the time that the entry was received by the Time Stamping Authority [14]. Let us now explore how we can use the aspects we have discussed to create a Proof of Action for each of an investigator's actions.

3.5 The Proof of Action

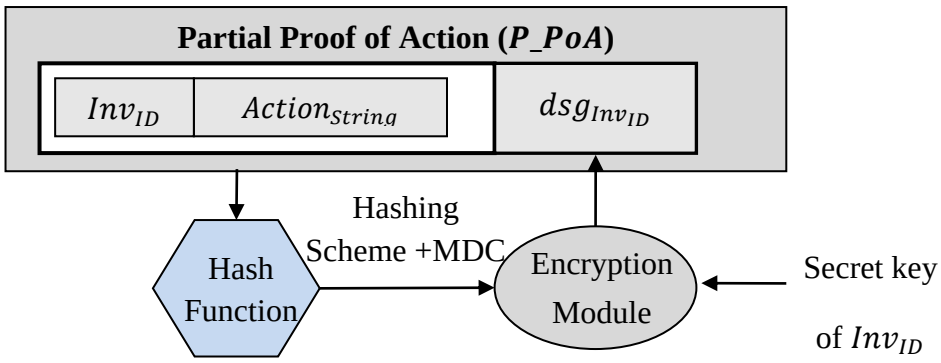
In order to create a Proof of Action we will use digital signatures and the Time Stamping Authority, in a similar manner to that of Coffey and Saidha [14] in their work on non-

repudiation between a sender and a receiver. The signature allows us to associate the investigator's identity with the actions they perform, and place a seal over this information, producing a partial proof (P_PoA). The signed information is then sent to the Time Stamping Authority, who transforms the partial proof to a Proof of Action (PoA) by appending their timestamp and signing the entire package. The notation used to describe how a Proof of Action is generated is also derived from the work of Coffey and Saidha [14]. An important aspect of their notation which will be used in this dissertation is:

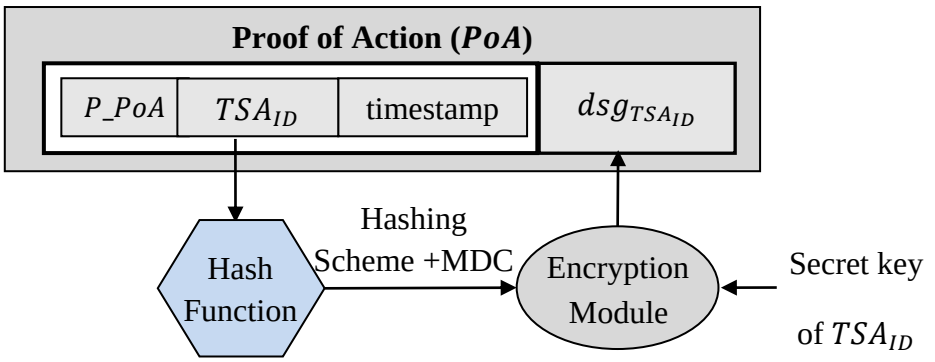
$[M]dsg_X$: which implies that the digital signature of entity X , in our case the investigator X, is computed for message M and then appended to the message.

Step 1: Partial Proof of Action (P_POA)

When an investigator performs a particular action to aid and make a contribution to the reasoning process, the system first retrieves the investigator's identity (Inv_{ID}). The system then generates a textual description for the action that took place (such as those discussed in Chapter 4), which includes detailed information about the action such as the tool that was executed, its version and so on, creating an $Action_{String}$. The system then appends the $Action_{String}$ to the Inv_{ID}, and calculates the Modification Detection Code (MDC) for the entire package using the system's hashing scheme. The MDC, along with information about the hashing scheme that was used, is then encrypted using the investigator's secret key, forming the signature for the action that was performed. The generated signature is then appended to the package and is called the Partial Proof of Action (P_PoA) [14].

Result	$P\_PoA = [Inv_{ID}, Action_{String}]\, dsg_{Inv_{ID}}$
Illustration of a P_POA	

Please note: The image is derived from the work and images of Coffey and Saidha [14].

<table>
<tr><th colspan="2">Step 2: Proof of Action (PoA)</th></tr>
<tr><td colspan="2">The Partial Proof of Action (P_PoA) that is generated is then sent to the Time Stamping Authority (TSA). The Time Stamping Authority then appends its identity (TSA_{ID}) and its timestamp to the P_PoA. The Modification Detection Code (MDC) is then calculated for this package, using the appropriate hashing scheme. The MDC, along with information regarding the hashing scheme that was used, is then encrypted using the secret key of the Time Stamping authority, forming a digital signature. The Time Stamping Authority then appends their digital signature and returns the result, called the Proof of Action (PoA), back to the system [14].</td></tr>
<tr><td>Result</td><td>$PoA = [P\_PoA, TSA_{ID}, timestamp]dsg_{TSA_{ID}}$</td></tr>
<tr><td>Illustration of a POA</td><td> 

Please note: The image is derived from the work and images of Coffey and Saidha [14].</td></tr>
</table>

The incorporation of Modification Detection Codes and Time Stamping Authorities helps us to easily verify the integrity of an investigator's contributions and their reasoning process. In this way we can place a seal over an investigator's work (and the time the various events occurred), ensuring that we accurately preserve an investigator's reasoning process. We can also verify the captured information by re-computing the Modification Detection Code. The inclusion of timestamps (and signatures) generated by a Time Stamping Authority and the use of Modification Detection Codes provides credibility for an investigator's work and actions that they followed, as well as for the time that these events occurred. The inclusion of timestamps aids in this regard, as should any modifications take place by the rightful owner of the contribution, then a new timestamp will be generated and signed. While it is important to note that key storage, management, protection and certificates is not the focus of this

dissertation, the Proof of Action plays a vital role as our first step towards capturing an investigator's reasoning process.

3.6 Chapter Summary

In investigations, an investigator and the actions they perform can impact evidence, the results of tools and, overall, the reasoning process that was followed. An investigator should therefore be accountable for such actions, and should not be able to deny their involvement in such actions. In communication protocols, one uses digital signatures to create proof of receipts and proof of origins, so that neither party can deny their involvement in the communication. We use a similar means to create a partial Proof of Action which allows us to undeniably associate the investigator's identity with their particular actions and to preserve the integrity of the information captured.

In order to ensure that we accurately record detailed information about these actions, a particular aspect of interest is capturing the time these actions took place. Investigators may use a variety of systems and devices that may use different formats for recording time and that may not be synchronized, which makes the interpretation of when actions and events took place challenging. By using consistent and accurate means for recording time in a uniform manner it allows us to construct a timeline of when actions were performed during the reasoning process, aiding the reconstruction of the reasoning process. The particular means we discussed for achieving this is through the use of a Time Stamping Authority. The Time Stamping Authority generates a timestamp when it receives a partial Proof of Action, and then digitally signs the entire package, producing a Proof of Action.

The Proof of Action relies on the use of hashing schemes, which generate Modification Detection Codes (also known as hashes). These Modification Detection Codes place a seal over the information captured in the Proof of Action, so that modifications can be easily detected. The incorporation of such a seal, the accurate time recording and the incorporation of non-repudiation helps to ensure that the actions we capture in the reasoning process cannot be easily challenged based on their integrity, and it helps to providing reliability to the captured reasoning process.

We are now ready to move onto Chapter 4, which will utilise the Proof of Actions to associate an investigator with the tools they use in an investigation, the actions they are used for and to capture the time these events take place. The seals incorporated in the Proof of Actions will also allow one to easily verify the integrity of the information.

Chapter 4

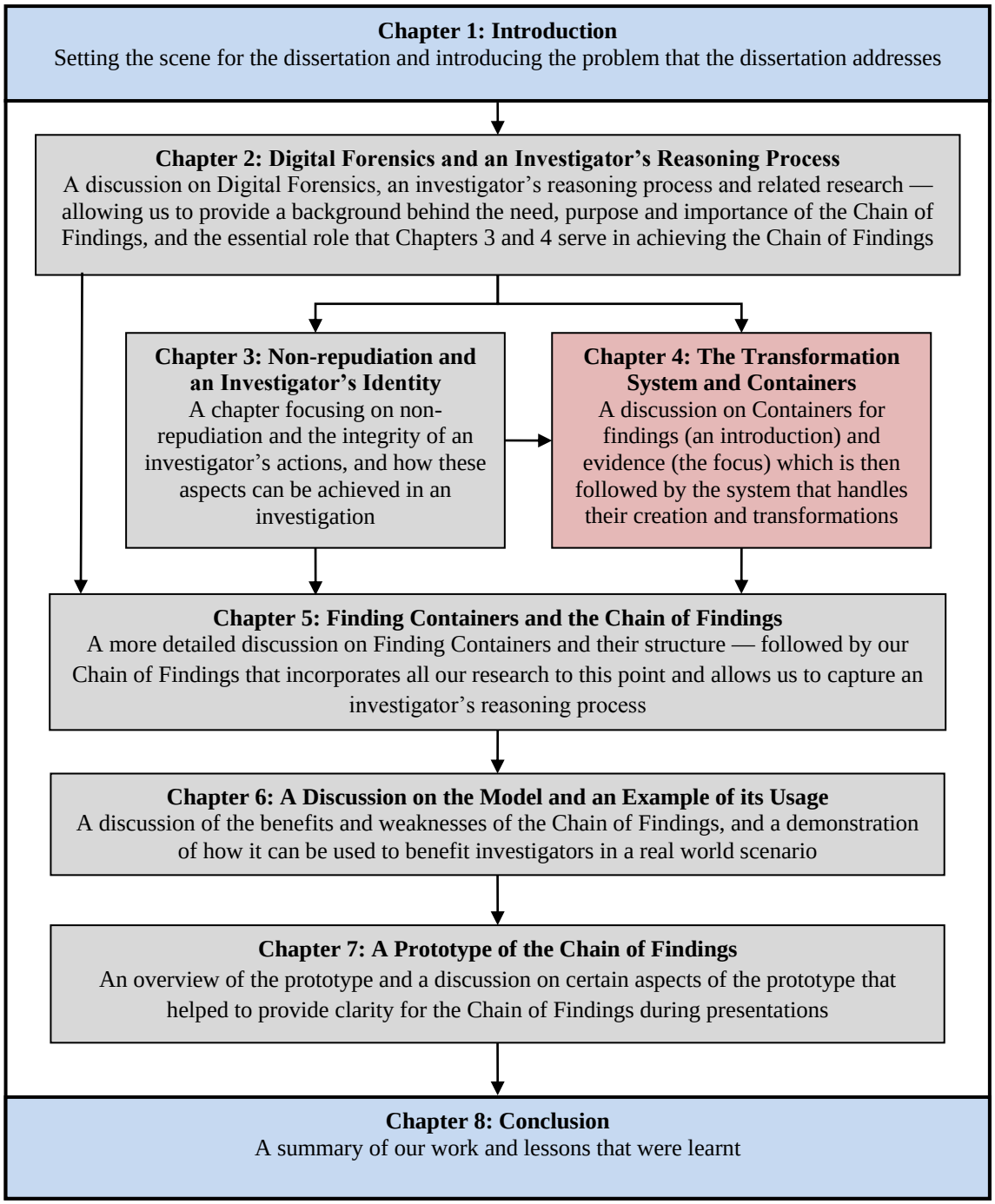
The Transformation System and Containers

Objectives

- To provide a means to manage and perform the transformations of an investigator's building blocks (evidence and findings)
- To provide a structure for an automated and detailed audit trail of transformations, and their reliability, using Proof of Actions (discussed in Chapter 3)

To the man who only has a hammer, everything he encounters begins to look like a nail
Abraham Maslow

A Map of our Current Location in the Dissertation



Chapter 4: The Transformation System and Containers

4.1 Introduction

In a Digital Forensic investigation, the use of tools helps an investigator to acquire, examine, analyze and otherwise find important information contained in the evidence, which may be relevant to the case at hand. Technology is developing at a rapid rate, as is the number of devices which may contain information relevant to an investigation. Unfortunately there are also increases in the volumes of data and heterogeneity of storage formats that these devices use for their storage medium (media) [45]. An investigator must therefore be able to keep up to date with these developments and formats, to ensure that they are equipped with sufficient tools and to perform their work in a forensically sound manner [32].

Turner [68][69] addresses this issue by creating a uniform wrapper for all types of digital evidence. This Digital Evidence Bag (DEB) allows evidence to be treated uniformly regardless of differences in source devices or file storage formats. These containers provide a uniform approach for encapsulating evidence and vital information regarding the evidence. This information includes attributes of the evidence and provenance information such as which Digital Forensic tools that have operated on the DEBs. The use of such a wrapper is beneficial as it allows all this information to be easily located and accessed. The DEBs then make use of multiple hashes, creating a form of layered hashing to provide multiple means for verifying the integrity of the DEB and its contents. The incorporation of these hashes and the detailed audit trails they contain, support the reliability of the evidence and, hence, the conclusions and arguments derived from them.

Schatz and Clark [59] highlight a number of shortcomings of the DEB. One particular shortcoming relates to the format and the vocabulary used for metadata and how it is syntax free and unspecified, hence relying on human interpretation. In order to provide a solution, they introduce the sealed digital evidence bag (SDEB), which makes use of Resource Description Framework and ontologies to provide a common representational format and to ensure that semantics are made explicit. Unfortunately, neither of these approaches cater for physical evidence, and hence an investigator cannot treat digital and physical evidence uniformly when forming their reasoning. These approaches also do not take into account non-repudiation, the reliability of the tools or the reliability of the contents of these wrappers. They also do not take into account the reliability of the tools used on these wrappers and their contents (after their creation), which may affect the contained evidence.

Each of the tools an investigator uses, whether it is for collection, extraction or analysis, can be used to produce new evidence. These tools may impact the evidence they are used upon. It is therefore important that the investigator follow the necessary legal requirements, and systematic process when using these tools [11]. The outputs of the evidence may also be used

as input into other tools in order to further transform the evidence into the required form — allowing errors in tools' output to propagate. The reliance on these tools in investigations means that any errors, inconsistencies or inaccurate results that these tools produce can drastically affect the investigation [24]. These tools can therefore impact the investigator's interpretation, the arguments they formulate and the conclusions they draw based upon the evidence.

In this chapter we introduce a Transformation System that creates an Evidence Container (section 4.3.1), allowing all evidence — digital and physical — to be treated uniformly. In our requirements for our Chain of Findings we stated that it should incorporate a means for investigators to reliably transform their building blocks, as is needed, and to record a detailed audit trail of the transformation and its reliability. During these transformations, the integrity of the source should always be preserved. The Chain of Findings should also allow for the capturing of provenance information regarding not only the transformations performed but also physical events, and it should ensure that non-repudiation is a core aspect of all recorded events. The Transformation System (section 4.3) handles all transformations of evidence, whether it is the transformation to a particular Evidence Container or the transformation of evidence into the required form, such as through analysis and reduction. Our intention is to ensure that these transformations can be relied upon, and to incorporate proof of the transformations' reliability into the provenance information that it generates for each Evidence Container it operates on.

In order to investigate how to ensure the reliability of these transformations and the Evidence Containers they produce, we investigate the guidelines for reliability specified in the Daubert Method (section 4.2). We then use these guidelines as a background to build upon the DEB and SDEB to create the Evidence Container, and to build upon an investigator's tools to create the transformations and the Transformation System, to ensure transformations and the Transformation System factor in reliability. The Transformation System uses Proof of Actions (Chapter 3) to store provenance information, ensuring that non-repudiation is an important aspect of the provenance information. The Transformation System also allows investigators to specify physical events regarding evidence and to store such provenance information. It is important to note that, while this chapter focuses on evidence and Evidence Containers, the Transformation System also produces and can operate on Finding Containers which are discussed in Chapter 5. These Finding Containers are Containers designed to capture an investigator's findings and their reasoning behind their findings. These Finding Containers resemble Evidence Containers, allowing them to reap the same benefits as Evidence Containers. The use of these Containers (Evidence and Finding based) allows an investigator to use evidence as well as existing findings uniformly as building blocks for more in-depth reasoning.

4.2 Reliability and the Daubert Method

The tools, techniques and procedures one uses to acquire and analyze evidence, and otherwise transform it into its intended form, may affect the admissibility of evidence [73].

Carrier [10] states that evidence must be shown to be relevant and reliable to be admissible. He discusses how one can show that evidence is indeed reliable by applying the Daubert method and hence complying with its guidelines. The Daubert method is a set of standards that provides guidelines which serve as a form of criteria for verifying and assessing the reliability of evidence and hence techniques, tools and their outputs [10][32]. The Daubert method helps to support the admissibility of evidence by showing that it was acquired in a forensically sound manner [10]. In our background we emphasized how accurate and reliable evidence is essential for drawing sound conclusions. The output of these tools may serve as evidence and may be used as building blocks for arguments, it is therefore important that one is able to verify the reliability of their building blocks and how they were produced, in order to help support the arguments that are derived from them. The guidelines that the Daubert method specifies to help achieve such reliability are testability, error rates, publication, acceptance, credibility and clarity [32].

4.2.1 Testability

Digital Evidence is easily damaged, altered and/or destroyed if one does not take precautions when handling and working with it [47]. In scenarios when an investigator must present their findings, they must ensure that they indeed took the necessary precautions to ensure that the evidence was properly handled. They must also ensure the accuracy of the evidence, to ensure that that the tools used can be trusted to not cause any alterations and damage the evidence in any manner [23].

Testability focuses on the empirical testing of a tool [32] in order to ensure the tool produces accurate and objective results [10]. Traditional means of testing are sufficient for testing for the presence of errors, but not for their absence [50]. Carrier [10] discusses how tests must be thorough so that they test for all particular conditions and circumstances which may occur, in order to strengthen the reliability of these tools in expert testimony. Carrier continues with how such tests help to ensure that relevant data is not missed or analyzed incorrectly. Two main types of tests must be performed, namely false negative tests and false positive tests. The false negative test is performed by ensuring that tools — such as imaging and extraction tools — provide all the data from the source device so that all of the source's contents are captured and retrieved. The false positive test ensures that a tool does not add additional data, which is not derived or otherwise extracted from the original source device, to the output they produce. Should one use image tools and they add additional data to the image they produce, then the image is not an exact duplicate copy of the original and, hence, the reliability of the image and any evidence derived from it, is drastically affected. Rowlingson [58] emphasizes the importance of hashing at several points throughout the investigation to ensure the integrity of the evidence and that they have not been altered, or incorrectly extended. Manson et al. [42] and Carrier [11] emphasize the importance of using secondary tools to help verify a tool's results and to help perform such tests.

A particular form of testing, calibration testing, is used to ensure that a tool is correctly configured, to verify the correctness of a tool (in its current working environment) and to test and measure the accuracy of its results [24][39]. Calibration is an important aspect when one

tests tools [45] as failing to do so may render digital evidence unusable and inadmissible and, hence, affect the defense of one's reasoning process in a court of law. It may also lead to an inaccurate conclusion [47] as well as failed legal action [22]. Calibration can help to ensure that the tools are correctly configured and tested for their intended environment before they are used. These configurations include setting a tool to use the correct file system and using the appropriate strategy for capturing network data based on the speed of the network, so that minimal data is lost. These configurations can also help to ensure that the tool is stable and that it is well suited to perform its intended task.

The stability of a tool is dependent on a number of factors including hardware, operating system, conflicting tools and processes [32], possible race conditions in the executing environment, drivers currently installed and the components involved in the execution of the tool [42]. Throughout the duration of the investigation, and the lifecycle of a tool — namely until they are disposed of or retired [48] — a number of changes can occur. These changes include configuration changes and components being changed in terms of infrastructure and/or changes to their programming such as replacements, disposals or alterations [22][29]. These changes of the configurations and the components, and the disposal of older, more outdated components, may affect the stability of a tool and its capability to perform its intended tasks. These impacts may affect the correctness of a tool, and it may also hinder their functionality (within the investigation) and hence their reliability.

Periodic calibration testing and testing of tools in general allows an investigator to discuss, convey [39], monitor [29] and verify the reliability of the tools used and also helps to verify that they worked correctly, accurately and objectively [39] under a variety of circumstances [24]. These tests and calibrations help further support that the results produced from the tools may be trusted [22], and that the tools perform their tasks in a forensically sound manner [24]. They also allow an investigator to use the tools with confidence, allowing others to see why such confidence in these tools exists. In addition, these tests can show that, should any errors or failures have occurred during the execution of the tool, they would have been reported and correctly captured [22], further improving the trust one can place in the results of these tools. It is important that the investigator document this entire testing procedure, so that they have a recording to show not only what tests were performed, and how they were performed, but also to capture the results of these tests.

Recording and capturing this information helps the investigator to support their claims of the correctness and reliability of the tools that they used [23]. These tools and their results form fundamental building blocks for the investigation and the investigator's arguments. During the course of the investigation, an investigator will have to use a number of tools to transform evidence into the necessary, usable form. It is this transformed evidence that will form the basis of their reasoning process. Ensuring that the basis for their arguments is reliable, well-tested, and well-documented provides reliability for their arguments, providing a more forensically sound foundation for them as a whole. Should an investigator maintain a history of such tests performed for each particular tool, the investigator can convey the reliability, correctness and the accuracy of the tools throughout the investigation, and through all the various changes which may have occurred. The recording of such a test history is beneficial

as it helps to indicate whether the investigator took the necessary precautions to help prevent any damage to the evidence, with regards to the tools they used.

Designing tests for Digital Forensic tools is a complex and time-consuming task [10][24]. Carrier [10] explains how traditional means for testing applications may be unable to detect all the bugs that reside in a tool's code as they are only able to detect bugs that they explicitly test for. Digital forensic tools require a stricter testing process, particularly due to their important role in evidence, the investigation and a suspect's life [10]. Time constraints make such detailed testing difficult and often make it challenging to take into account all possible conditions and situations. It may also be difficult for the investigator to create tests that take into account all the necessary circumstances, perform all the necessary checks and are thorough enough, particularly due to the lack of a formalized test methodology [10]. Should the tests fail to do so, they may fail to show the correctness and reliability of these tools, and they may not be able to withstand scrutiny against them in a court of law [47]. Documenting these tests may also place further time-consuming, administrative burdens on the investigator [58]. This problem is further amplified by the fact that an investigator will have to repeat this work for each tool they may require in the investigation. What if these tasks were handled on behalf of the investigator? What if the investigator had a particular means to store, test, calibrate, update and overall manage their tools, on their behalf, in a reliable manner?

4.2.2 Error Rates

The second guideline discussed in the Daubert method, relates to error rates of a tool [10][11] and how incorporating this information can help to establish the reliability and accuracy of the tool [13]. There are two main types of tool errors, namely Tool Implementation Errors and Abstraction Errors. The Tool Implementation Error is a result of bugs in the code of the tool and the developer(s) not fully understanding specification or otherwise using an incorrect specification [10]. Tools rely on their implementation of a specification in order to correctly interpret it, so that they can analyze, interpret and utilize formats based upon it. Incorrect interpretations may affect the results these tools produce and the reliability one can place in them. Unfortunately, certain formats and their associated specifications are proprietary and tools may therefore suffer as these specifications are not publically available and hence not well-understood by the general public [11]. Carrier emphasizes that the history of these bugs and their severity can provide a means for measuring a tool's error rate. Unfortunately, closed-source software is unlikely to follow such an approach due to a potential loss of revenue and damage to the developer's image [10]. The second type of error that Carrier discusses is Abstraction Errors.

An Abstraction Error, according to Carrier [11], is the result of a tool performing lossy abstractions, or conversions, from an input to an output without having a 100% certainty. This form of lossy conversions is common in many-to-one scenarios, where multiple inputs can result in the same output. Examples of Abstraction Errors include IDSs, file carving techniques and the recovery of deleted files. An IDS (Intrusion Detection System) reduces multiple packets into a particular attack — however, the system is not 100% certain (as Carrier [10][11] puts it) that the packets are indeed part of the attack and, hence, introduces a

margin of error and uncertainty [13]. Similarly, in file carving techniques where a bag of bits is carved to reveal various files and directories within them, a tool may introduce a margin of error as it lacks 100% certainty that the bytes are meant to be interpreted and extracted in a certain manner [15]. Tools that help to recover deleted files may suffer similar problems, especially due to a lack of publication and specification and, hence, a lack of a standardized manner for performing such a task [10]. In each of these scenarios there may be a number of possible manners for interpreting the bytes [15] and the tool must use its programmed knowledge to make this interpretation. Carrier discusses how each tool can be easily given an Abstraction Error value, and the accuracy of the value can be improved with research [10].

The maintenance of a test history for each particular tool can aid in this regard by maintaining test and error information for each of the tests performed, across each of the versions of the particular tool. The history can therefore help to provide more in-depth information about a tool's errors and error rates and, hence, its reliability over its life cycle.

4.2.3 Acceptance and Publication

The acceptance and the publication of a tool (and the procedure it uses) help to ensure that the tool has undergone peer review, and they are important for establishing the reliability one can place in the tool and the evidence it produces [10]. Peer review helps to ensure that any critiques and flaws will have been discussed, challenged and defended, helping to establish the acceptance of a tool and its process in the Digital Forensic community [32]. Carrier [10] emphasizes that procedural details of a tool should be published in a language other than just the tool's source code language.

Publishing a tool's source code, along with the associated procedural documentation, allows an investigator and external parties to verify that a tool does follow the process that is specified by the tool [10]. The FBI's forensic journal go on to state how the code of digital forensic software may be requested and analyzed in order to determine the admissibility of evidence and the manner in which it was acquired and produced [10]. Closed source and proprietary forensic tools make access to the source code difficult, if not impossible [10][11][36][42]. It is therefore important that an investigator take this into account when they choose the tools which they will use in an investigation [10]. Capturing the various operations performed by a tool on a particular piece of evidence allows these operations to be more easily analyzed and possibly cross-referenced against existing publications, further supporting the reliability of a particular tool.

4.2.4 Credibility

Jansen et al. [32] discuss the importance of an investigator's credibility and the importance it plays in the reliability and accuracy of their results. An investigator must have the necessary skills, training and expertise [73] to use a tool correctly and to not spoil (damage or alter) the evidence when using the tool [22]. These qualities help an investigator to correctly interpret the tool's results, to understand the tool and its limitations and to avoid errors in the tool's use [15]. What if an investigator was only able to use a particular tool should they have the necessary skills, training and experience to do so?

4.2.5 Clarity

Jansen et al. [32] discuss how clarity is assumed to be implicitly incorporated in the Daubert method. It is an important guideline and refers to whether a tool's technique and results can be explained with sufficient detail and simplicity so that it can be correctly understood by its intended audience. Turner [67] supports this and discusses how the provenance of a piece of evidence should be succinct and clear. Clarity plays an essential role in the presentation and reporting phase of an investigation so that the reliability one can place in the evidence can be easily conveyed. Ensuring clarity with regards to an investigator's audit trail can benefit the reconstruction of an investigator's results and their findings as well.

Throughout the dissertation and this chapter we have emphasized how an investigator should maintain an audit trail of all their actions they have performed with regards to digital evidence. The National Institute of Justice [47] go further and state that a complete, accurate, detailed and comprehensive report of all the actions taken by the investigator should be fully documented and reported. They discuss how this applies to their findings they may discover and their results they produce and present. The Proposed Standards for the Exchange of Digital Evidence discuss how this documentation must be preserved and available for review [32]. The Association of Chief Police Officers [73] state that the audit trail must be captured in such a manner that it can be easily repeated by a third party, such that they will arrive at the same results. The benefits of such reconstruction is that it greatly improves the reliability one can place in an investigator's results. Ensuring that the audit trail is clear and easily understandable can greatly aid the reconstruction process. Let us now explore the Transformation System which incorporates these guidelines and captures these audit trails for evidence.

4.3 Transformation System

The Daubert method provides a useful background, in terms of guidelines, for the Transformation System, to help ensure that all transformations it performs can be relied on and that sufficient facilities and documentation are provided to support such reliability. The Transformation System that we introduce is composed of three main aspects, namely Containers (Findings and Evidence based), Transformations (section 4.3.2), and the Transformation Manager (section 4.3.3). The Containers consist of Evidence Containers (section 4.3.1) that form the foundation for an investigator's arguments and Finding Containers that encapsulate these arguments. These Containers incorporate all associated metadata and provenance information that help provide a detailed audit trail. In the remainder of this chapter we focus on Evidence Containers that allow investigators to treat physical and digital evidence uniformly. In this manner both types of evidence can be used uniformly in an investigator's reasoning. Chapter 5 focuses on Finding Containers. In order to derive the Evidence Container, we build on the Digital Evidence Bag (DEB) discussed by Turner [68][69] and the Sealed Digital Evidence Bag (SDEB) discussed by Schatz and Clark [59].

Transformations are derived from the idea of DEB and SDEB applications and are the tools that are used for creating these Containers and transforming them into the desired format, in a

reliable manner. Transformations that work with digital evidence serve as a metaphor for Digital Forensic tools, which are used to transform evidence into its intended form, such as through extraction, analysis or by transforming the evidence into a viewable format for the investigator. Transformations that work with Finding Containers are discussed in more detail in Chapter 5. An investigator may require a number of these transformations, finding and evidence based, during an investigation. For example with regards to digital evidence, a number of transformations are required, particularly due to the number of file storage formats that may exist on a particular digital source at any particular time, and the growing volume of digital media and devices. The Transformation Manager is responsible for storing all these transformations, ensuring that they work correctly and that they are up to date, as well as making these transformations easily accessible. The Transformation Manager also incorporates the means to register new transformations, so that the Transformation System is flexible and can keep up with (new) developments in digital devices and media.

4.3.1 Evidence Containers

There exists a large number of digital records that may each play a role in an investigation, ranging from documents on a computer and telephone contact lists to network traffic patterns [54]. Raghavan et al. [54] discuss how investigators encounter a number of challenges when using these records due to a number of reasons. These reasons include new electronic devices and media [59], new operating systems, new storage formats, new communication protocols, and the growing volume of digital media [68]. Coping with such challenges is difficult, as investigators require a large array of specialised tools for capturing, extracting and analysing data from the growing number of digital devices [68] and the diverse data they may contain [54]. Schatz and Clark [59] go on to discuss how the large number of acquisition and analysis tools use a variety of ad-hoc and proprietary formats for storing evidence content, analysis results and metadata regarding evidence. Schatz and Clark discuss how the conversion between the formats is complicated, time-consuming and how it may result in incorrect evidence data and/or the loss of metadata. Furthermore, they discuss how the validation of results is difficult due to the lack of standardization and a common representational format for metadata.

Despite these difficulties, an investigator must take these pieces of evidence into account, in their entirety, in order to form a timeline of events and to form a complete picture of the incident to draw sound conclusions (Chapter 2 [47]). There is therefore a need for capturing, understanding, and analyzing information from disparate digital sources uniformly [54], as well as their associated metadata [20][37]. The Forensic Integration Architecture (FIA), by Raghavan et al. [54], provides a framework that allows investigators to integrate evidence and its associated information uniformly, independent of source and storage formats (Chapter 2). The FIA allows for the registration of interfaces and interpreters to ensure that it is able to keep up with future types of evidence. However, the FIA does not factor in provenance information, such as the applications used on the evidence and how the evidence came to be, potentially affecting the reliability of evidence. The Advanced Forensics Format (AFF) provides a publically-disclosed format for storing acquisition related metadata along with a disk image [59]. However, the AFF caters only for hard disc images [54] and the metadata,

which is represented using name/value pairs, provides no means for attaching semantics — hence relying on human interpretation [59].

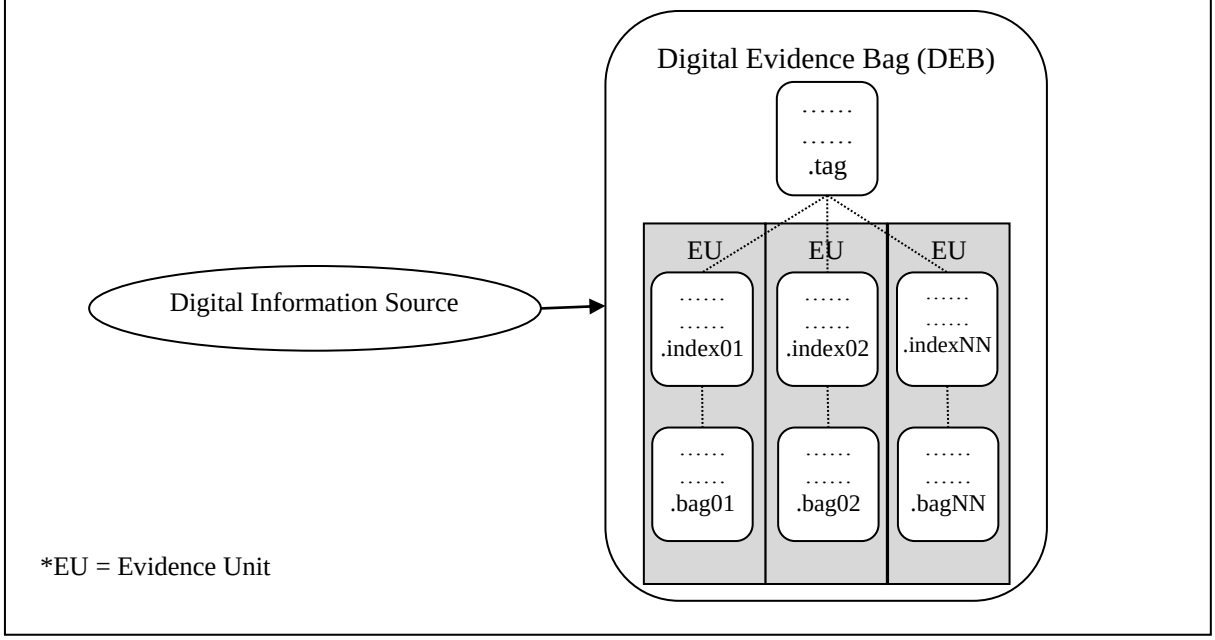
Turner [68] continues with the financial investments and time that has gone into creating and formulating existing tools and techniques. Turner therefore encourages researchers to pursue approaches that allow the reuse of existing tools and techniques. He pursues such an approach, which he calls the Digital Evidence Bag (DEB). These DEBs provide a uniform wrapper for all types of digital evidence (and their metadata) in an investigation, and they incorporate a detailed audit trail of all processes performed on the evidence as well as integrity checking [69]. The DEB is derived from a physical evidence bag — which is used in traditional evidence capture — and Turner uses this metaphor to more easily communicate the function of the DEB [59][68]. In section 4.3.1.2 we discuss the SDEB by Schatz and Clark [59] which builds upon the DEB to add semantic information to the contained metadata and to address other drawbacks of the DEB.

4.3.1.1 The Digital Evidence Bag (DEB)

In law enforcement, physical evidence bags are used to store and seal evidence that is considered relevant to the investigation [68]. These bags use a seal number for identification purposes, an optional tamper evident tape closure seal [59], and they have a tag attached to them [68]. The seal helps to verify the integrity of the collected evidence by reflecting whether any tampering may have taken place. The tag contains information about the particular piece of evidence, such as provenance information [59] and a description of the particular piece of evidence [68]. Each of these physical evidence bags may vary in size, shape and type, depending on the contents they must contain, but the concept of the evidence bag and the associated tag nevertheless allow these contents to be treated uniformly [68]. Turner builds on this method to provide a similar approach for the digital environment, providing a uniform container for digital evidence.

A DEB is a universal container [69] that provides a wrapper for any type of digital information or digital evidence [68] from any digital source [69]. The DEB not only contains the digital evidence, integrity information, evidence metadata [59] and information about how the evidence came to be but also which applications were used by the investigator to analyse the DEB and its contents [68][69]. While Turner mainly discusses the benefits of such an approach with regards to imaging large volume digital media [68][69], he also elaborates on further benefits for digital forensic tools. Examination and analysis applications can benefit from such a uniform approach as they can operate on the DEBs and be independent of the many potential source devices and media that the DEB could be created from. The DEB (demonstrated in Figure 3) is composed of three types of files, namely bag files, index files and a tag file [59][68][69].

Figure 3: The Digital Evidence Bag (DEB) [68][69]



The bag file contains the actual evidence that is captured, such as file system images, network traces, the contents of image files [59], or files that fall into a particular category [68]. The index file contains all details of the corresponding bag file [59], such as file and folder names, last access time or details of the physical device, such as make, model and serial. Turner calls the pairing between the bag file and its corresponding index file, an Evidence Unit [59]. These Evidence Units each include an integrity hash for both their index file and the bag file they contain [69]. Turner [69] discusses how, when performing imaging, the first Evidence Unit is reserved for case notes (such as photographs, sketches of the scene, notes from interviews) and case-associated metadata at the time the DEB is created. These details include information such as the imager used, a hash of the imager application and its configuration file. The format definition for the index file is specified in the tag file [68] by a sequence of meta tag labels [69]. The customisable nature provided by this sequence of meta tag labels allows the DEB to store information from a wide range of devices and media. In Chapter 2 we elaborated on how digital evidence is circumstantial and how it must be combined with physical evidence. Building on Turner's idea, one could create Evidence Units for physical evidence, which would reference the physical evidence in storage, contain metadata about the physical evidence and possibly pictures and sketches of the physical evidence. For example, a cell phone found at the scene and information about how it was collected. In this manner physical (the cell phone) and digital evidence (such as the contents of the device) can be treated uniformly by an investigator using Evidence Units. Let us now discuss the tag file.

The tag file for the DEB is specified using unstructured text [59] and contains information such as the DEB reference identifier, information about the evidence contained in the DEB, namely the Evidence Units, and the date and time the capture process started. The tag also contains a hash of the captured information in the DEB and a tag seal number [68][69]. The

tag seal number is used to uniquely identify the DEB and consists of the hash of the tag file to date and hence changes whenever the contents of the DEB change. The tag file also contains Tag Continuity Blocks, containing all provenance information of when a DEB application accesses the DEB. Whenever a DEB application accesses a DEB, it updates the tag file with a Tag Continuity Block (TCB). The TCB reflects the history of operations that were performed on the particular bag file, the timestamp of when the application accessed the DEB, and it includes the application function and signature [69]. A large amount of information regarding the evidence containers and particular applications and processes is therefore recorded along with the evidence, in a single location — making it more easily accessible. These applications then update the tag seal number accordingly.

The importance of a detailed audit trail has been emphasized throughout Chapter 2 and earlier in this chapter. We have also discussed how the audit trail should be recorded such that it is clear and concise, unambiguous and comprehensible by its intended audience. In addition, we emphasized how it should be recorded in sufficient detail and in such a manner that it is easily repeatable by a third party. By maintaining a uniform approach for recording events and the time they took place, regardless of tools used, it helps maintain a more accurate timeline of events and it helps aid in reconstruction. Tan [64] discusses how by using a uniform format for the recording of events can greatly aid in their interpretation. Turner goes on to discuss the benefits with regards to interpretation of such information and its validation, should the nomenclature used for recording such information be consistent [67].

In Tan's [64] research he discusses how an investigator must be able to defend the tools they use and reason about its methods. An investigator may therefore be required in expert testimony to describe in detail how a Digital Forensic tool performed its task [15]. The potential number of versions and updates [10][32][42] may however make it difficult for an investigator to accurately keep track of and explain the exact steps a digital forensic tool took. Ensuring that each DEB application records its operations within a TCB in detail can greatly aid the investigator and their expert testimony. Recording the sequence of operations that a DEB application follows to perform its particular task can further support the reliability of the tool. For example, should the sequence of operations (or a subset thereof) be well accepted or published in the Digital Forensics field, it can help to support the reliability of a tool and its results. Since each of these TCBs are added to the tag file in the order they are performed, not only can the operations be captured but also the sequence the DEB applications were executed in. However, this only captures the digital events of a DEB. Incorporating a means that allows investigators to add TCBs to capture non-digital provenance events for evidence (such as with regards to transportation) can help ensure that a more thorough Chain of Custody for evidence is maintained in a single location. The use of multiple hashes within the DEB and its layered approach provide a stronger form of association between the DEB and its contents, and they provide multiple means to verify the integrity of the DEB and all the information it contains.

The format for these TCBs is however not defined by Turner, nor does he discuss how the operations are recorded — whether they use source code, pseudocode or perhaps another approach. Turner also does not discuss the recording of the investigator responsible for

performing these operations, the format used for these recordings and he does not discuss a means for ensuring non-repudiation. In addition, Turner does not discuss whether information regarding the application's reliability is recorded, such as whether they were tested or calibrated before use, or the results of such tests. Furthermore, Turner does not discuss whether the generated timestamps use synchronized time and whether they use a uniform format. Ahmad et al. [3] state that combining multiple log records that contain information about a single event can increase the comprehension of these logs and reduce the number and volume of these log files. In Chapter 3 we introduced the Proof of Action and how it can capture accurate timestamp information, the investigator's ID and the actions performed in a manner that is not easily refutable. By using such an approach, and recording the test history for these applications and the sequence of operations within these Proof of Actions using a uniform format, a more detailed and reliable TCB can be created. The sequence of these Proof of Actions can then be used to provide a more thorough history of such actions, forming a Proof of Action History, benefiting reconstruction and providing a stronger base to circumvent challenges against evidence. Recording the rationale behind these actions within the Proof of Actions can also be useful as it allows investigators to capture the reasoning behind why certain actions were performed and why they were performed in the manner they were — further benefitting these TCBs.

The TCBs discussed by Turner also do not seem to capture proof that the evidence and the DEB itself has not been changed or otherwise altered during the DEB application's execution. This is particularly so as integrity hashes for the Evidence Unit and its contents do not seem to be recomputed, retested or otherwise verified after a DEB application executes — a TCB is merely added to the tag file, and a new tag seal is generated. In Chapter 2 we discussed how digital evidence is subject to the same rules and laws that apply to documentary evidence and how the investigator must show that the evidence presented is no more or no less that when it was first taken into possession.

Operating systems, simply by being started, and various other programs during execution can make changes, with or without an investigator's knowledge or permission [73]. These concerns are problematic in a digital investigation. This is amplified by the fragile nature of digital evidence and how improper handling and actions can result in its spoliation. An investigator must therefore follow special precautions to preserve the integrity of evidence and use accepted forensic procedures [47]. The failure to do so may render evidence unusable, inadmissible and may result in inaccurate conclusions. In Chapter 2 we discussed how the *fragile nature of digital evidence and its various sensitivities encourage an investigator to perform regular integrity checks on the evidence, so that they can confirm and verify the evidence's integrity and reliability throughout the investigation [4][47]* (Chapter 2). These regular integrity checks therefore provide an important means for verifying and ensuring evidence is no more and no less than when it was first collected. Ensuring that each DEB application performs and records various precondition and postcondition tests (and results), such as testing and verifying a DEB's integrity hash and all its contents, can help provide greater reliability to the DEB and its contents as a whole. These precondition and postcondition tests and their results can then be recorded within the generated Proof of

Action, allowing each Proof of Action to provide greater information regarding the DEB application in a single, easily accessible location.

Schatz and Clark [59] highlight a number of further problems with regards to the DEB proposed by Turner. The format and the vocabulary used in the tag and index files is syntax free and unspecified [59]. Because of this, the provenance information and evidence metadata they contain relies on human interpretation and may hence suffer as a result. While Turner states that the DEB is able to contain other DEBs, differentiating it from monolithic formats in use [67], Schatz and Clark [59] highlight another concern. Schatz and Clark elaborate on how Turner does not define a scheme for referencing evidence and metadata between DEBs and this hinders the capability of forming a corpus of DEBs. Schatz and Clark attempt to solve these highlighted problems with their sealed digital evidence bag (SDEB).

4.3.1.2 The Sealed Digital Evidence Bag (SDEB)

The sealed digital evidence bag (or SDEB [54]), proposed by Schatz and Clark [59], is immutable, and builds upon the DEB to provide a means for constructing a corpus of evidence. The SDEBs use a Uniform Resource Name called the Digital Evidence Identifier (DEID), which is derived from the Life Sciences Identifier (LSID), to globally identify SDEBs and metadata instances. A corpus can be created by embedding SDEBs within SDEBs and by SDEBs referencing other SDEBs. Schatz and Clark elaborate on how there is no common representational format for evidence metadata. They discuss how practitioners and researchers in digital forensics do not use standard terminology, which is what Schatz and Clark attempt to address with their SDEB.

The lack of a common representational format and standard terminology can result in a number of issues, making the interpretation of metadata information difficult. Previously (section 4.3.1) we discussed *how an investigator must take into account evidence in its entirety in order to form a timeline of events and to form a complete picture of the incident to draw more sound conclusions*. We then continued with how there is hence *a need for capturing, understanding, and analyzing information from disparate digital sources uniformly [54], as well as their associated metadata [20][37],* to achieve this. We would also like to allow investigators to not only uniformly capture their findings and the evidence that supports them, but also detailed information about why they do so. In our requirements for the Chain of Findings we discussed how investigators must also be able to capture the properties (attributes) of the evidence and the criteria that they comply with, that overall allows the evidence to serve its role in the investigator's reasoning. These attributes are contained within the metadata, and should interpretation be an issue, it can affect the interpretation of the captured findings as a whole.

The problem of bringing together heterogeneous and distributed computer systems is called the interoperability problem by Wache et al. [70]. In their discussion they elaborate on two particular forms of heterogeneity that make such interoperability difficult, namely syntactic and semantic heterogeneity. Syntactic heterogeneity is a result of using different data types and representations [62], formats [6] and/or structures [70] for storing data [6]. Semantic heterogeneity refers to the contents of an information item and its intended meaning [70], and

it may occur due to a number of reasons including precision conflicts, scaling conflicts, and name conflicts [62]. Scaling conflicts can occur due to the use of different reference systems (such as different currencies) for information items [70]. Naming conflicts include synonyms and homonyms [62]. Two attributes are referred to as synonyms when they have different names but have the same semantic meaning [33][62]. Homonyms occur when two attributes are semantically unrelated but have the same name [62]. Schatz and Clark [59] use ontologies to overcome these terminological and representational problems in the Digital Forensics field.

An ontology, defined by Gruber, is a formal and explicit specification of a conceptualisation [6][59][70]. Bernard et al.[6] go into detail to describe the definition by elaborating on what is meant by conceptualisation, explicit specification and the use of the term formal. They begin by discussing conceptualisation and how it refers to an abstract model of how people think about the real thing in the world. They then discuss explicit specification, and how it refers to all concepts and relations of the abstract model being given explicit names and definitions. Bernard et al. then discuss formal, and how it refers to terms being defined using a formal language with well-understood properties. Bernard et al. discuss how the use of such a formal language helps to prevent any ambiguity with terms used. Kashyap and Sheth [33] describe how ontologies can themselves be seen as metadata, which provide a vocabulary of terms that can be used for constructing more domain-specific metadata descriptions. Schatz and Clark [59] state how an ontology is a means of conceptualising a domain of discourse in terms of concepts, properties of concepts, restrictions on properties (including permissible values for these properties [49]), and relationships of entities. Noy and McGuinness [49] refer to these concepts in a domain as classes, and they discuss how the design of these classes differs from those used in object-orientated code. They discuss how classes in object-orientated code focus on methods and operational properties whereas classes in ontologies focus on structural properties. They continue with how an ontology, together with instances of its classes, combine to form a knowledge base.

Overall, the benefits of pursuing an ontology-based approach is that the semantics are described explicitly [70], sharing a common understanding of the information [59], rather than implicitly, relying on human interpretation [6][59]. Raskin et al. supports and encourages the use of ontology based approaches in the security field, in order to organise and unify terminology and nomenclature [59]. Schatz and Clark [59] use the Resource Description Framework (RDF) to provide a common data representation layer for digital evidence and related metadata in their SDEBs. They then use ontologies to describe the vocabulary that is related to this data. In this manner they help to overcome these particular terminological and representational problems. The use of such an approach allows one to explicitly define metadata, or rather metadata classes, capturing important information including semantics, properties and relationships. One can then construct the necessary metadata from instances of these classes, allowing them to inherit and comply with the explicit specification from the class they are derived, while also ensuring consistency among different instances of the same class. While the SDEB resembles the DEB in structure, it does have some differences — but before we delve into them we first elaborate on the types of metadata that is incorporated within the SDEB.

Metadata is often seen simply as data about data [57]. Sheth and Kashyap [33][63] however go further and describe it as information about data. Schatz and Clark [59] refer to the actual evidence contained within the SDEB as the evidence content and they discuss three main types of metadata contained within the SDEB namely evidence metadata, provenance metadata and integrity metadata. Evidence metadata refers to metadata that is based on or otherwise related to the evidence content, such as the file path or the last modification date. This form of metadata can therefore be seen as storing characteristics of the evidence content. We refer to this form of metadata as *attributes of the evidence* in our model. Schatz and Clark continue with how provenance metadata refers to the provenance of the evidence and how integrity metadata is used to verify the integrity of the evidence content. The integrity metadata they discuss is, however, based on evidence content, and we therefore regard the hash of the evidence content as an attribute of the evidence. We do so to avoid confusion between the integrity of the evidence content (which is achieved through various message digests) and the integrity of the entire bag (which is achieved through a particular file called, the Tag Integrity File).

The SDEB maintains integrity information regarding the tag's contents outside of the tag file in a file called the Tag Integrity File. This approach differs from Turner's and is mainly due to implementation issues, but it nevertheless serves as a useful flag. SDEBs remain unsealed and mutable until the particular Tag Integrity File has been added to the SDEB, and hence the absence/presence of Tag Integrity File can be used to determine whether information can still be added to the SDEB. The immutable nature of SDEBs means that they provide no construct for the Tag Continuity Blocks proposed in the DEB. Schatz and Clark [59] expect the application of tools on SDEBs to result in further SDEBs being produced. They continue with how the provenance metadata of these newly created bags would then contain detailed information of the tools used, along with a reference to the original, and would therefore serve the same role as Tag Continuity Block. In this manner the SDEBs remain immutable and one can easily determine the source from which the secondary evidence was created, without data redundancy.

The problem with such an approach is that it makes it difficult to easily gather provenance information regarding a particular piece of evidence without knowledge of all secondary evidence derived from it. Another problem with this approach, is when an investigator uses a tool to view the evidence content of a SDEB (such as a JPEG or Excel spreadsheet). These viewing activities will however not result in a new SDEB being created. Furthermore, should these activities not be recorded, it results in only a partial provenance being maintained for a SDEB, affecting the completeness and reliability of the captured provenance information. In addition, the use of the approach discussed by Schatz and Clark for capturing provenance information makes it difficult to easily add provenance information regarding non-digital events. Embedding however has an important use, as discussed by Schatz and Clark, and that is in aiding investigators to provide additional information regarding evidence.

Schatz and Clark [59] discuss how an investigator can add additional information about evidence through the use of embedding, while keeping the SDEB unmodified. They provide an example to elaborate on how this can be achieved. In the example, they describe a

simplistic imager for a hard drive that produces an immutable SDEB. Should one wish to add details such as the serial number on the drive or the investigator that performed the imaging, one can use a SDEB annotation program. Such an application creates a new unsealed digital evidence bag, along with a new tag file which is then used to store the additional information through the use of the annotation program's interface. Once all additional information has been added, the bag is sealed. It is important to note that the annotation program avoids creating a new Evidence Metadata File (referred to as the index file in Turner's model) for this new bag as no new evidence is created [59]. The use of such an embedding approach allows the original SDEB to remain unmodified while still providing the flexibility for investigators to add additional information regarding the evidence. A similar approach can be used to allow investigators to add provenance information for non-digital events. The drawback of such an approach is that it will require an additional SDEB being created each time, possibility resulting in a number of such SDEBs being created. A further drawback is that this additional provenance information will not be recorded along with provenance information regarding digital events. Pursuing such an approach may hence complicate the task of determining how evidence came to be in its current form.

The fact that the tag file can also be used to store attribute information regarding the evidence content (such as the serial of the drive from which the image belongs) means that an investigator may have to search not only the Evidence Metadata Files (referred to as index files in Turner's model) but also the tag files of SDEBs for attribute information. Similarly, when one forms a corpus of related evidence, such as a group of related SDEBs regarding files of interest, while no evidence is created, an investigator may want to add attributes regarding the corpus. These attributes will then appear in the tag file (possibly in a new SDEB). These differences in storing metadata may make interoperability among SDEBs difficult. SDEBs may hence require a more in-depth examination in order to accurately retrieve and interpret their contained information.

When an investigator adds additional information to a SDEB and this SDEB was already used to construct a corpus of evidence, the corpus may have to be updated to contain the new SDEB. Unfortunately that may not be possible as the SDEB containing the corpus may already be sealed and a new corpus may need to be created using a new SDEB. There is another drawback which may also affect these corpuses of evidence, related to the admissibility of evidence. Schatz and Clark [59] do not discuss the admissibility of evidence or where such an attribute would be recorded — the index or tag file. The dynamic aspects of an investigation could mean evidence may no longer be admissible and such an attribute would require modification. The use of embedding could aid in this manner because investigators can embed an SDEB containing the incorrect attribute value within a newly created SDEB. Such a modification may also require changes to corpuses of evidence. In summary, the use of such embedding can result in a large chain of SDEBs, where each SDEB contains additional information stored using a non-homogenous approach. Should an investigator use an SDEB as a building block for their reasoning, they would have to (regularly) check for such a chain of SDEBs involving their SDEB. The investigator would have to do perform such a check in order to determine whether there are any changes

regarding their SDEB and to factor these changes (as well as the impacts of these changes) into their reasoning. This problem is amplified should multiple investigators be involved in the same investigation.

Embedding has another benefit tough, namely that it can be used to embed the results of a secondary tool that is used for verification purposes. In this manner a single location, namely a SDEB, can be used to contain provenance information regarding evidence, how it came to be, the source it was derived from and evidence metadata, as well as verification information from a secondary tool. Embedding can also allow an investigator to embed evidence collected at the scene regarding a particular digital device in a single location. For example, the manuals collected at the scene for a particular cell phone can be embedded into the particular SDEB regarding the cell phone. The use of references in these SDEBs has its benefits as well, and by building upon Turner's work on DEBs, these SDEBs can also provide a useful means for storing physical evidence.

Previously we discussed how we could create an Evidence Unit for physical evidence. Using the SDEB and its associated referencing with its unique identifiers allows us to not only capture the digital evidence, but also the physical evidence from which it came. In this manner one can directly link digital evidence, and hence all evidence derived from it, with the physical evidence. An investigator can then link a particular cell phone found at the scene with information found on the device, which may then be linked to the results of the analysis of such information. Secondary evidence derived from the collected physical evidence may also benefit from such an approach, allowing it to be treated uniformly with digital evidence and allowing it to maintain information from where it came. An investigator can then link fingerprints found on the device to the cell phone, which can then be linked to the results of fingerprint analysis. These SDEBs achieve referencing and embedding with the use of the tag file.

Using the tag file for embedding and referencing makes sense in some scenarios, such as adding of verification results or capturing the source from which evidence came, but in others it does not. For example, when an investigator forms a corpus of evidence, then the actual content of a SDEB will be stored in the tag file rather than in the Evidence Content File (referred to as a bag file in Turner's model). This will result in an inconsistent manner for storing the content of SDEBs. Furthermore, should an investigator wish to add attributes to such a corpus, problems previously discussed will resurface. Schatz and Clark [59] unfortunately do not elaborate on this matter or on how such a corpus is handled. Now that we have investigated the DEB and the SDEB, let us introduce the Evidence Container that builds on them and attempts to overcome their drawbacks.

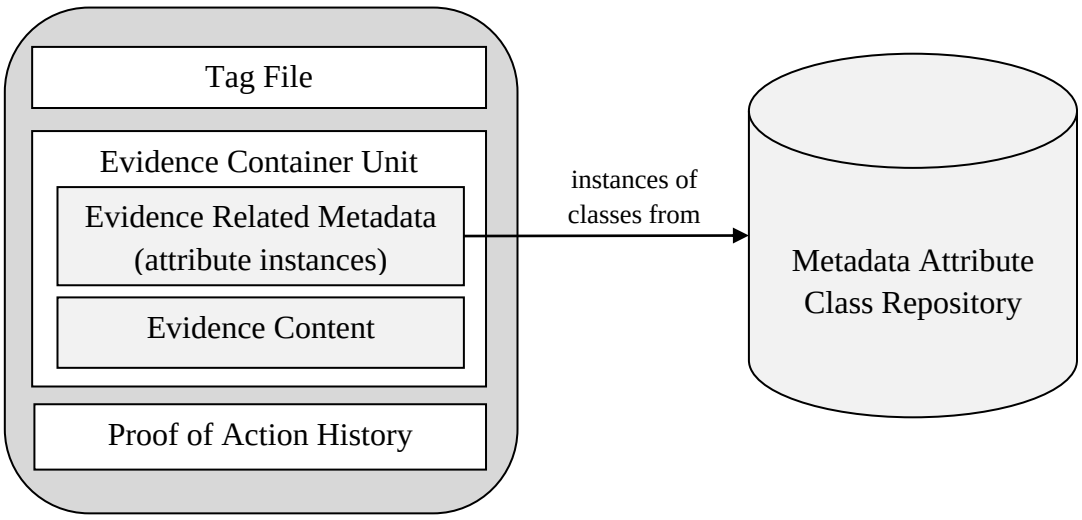
4.3.1.3 The Evidence Container

We build on the DEB and the SDEB and incorporate the aspects we have previously discussed in order to create a uniform Evidence Container for digital and physical evidence, which can make use of referencing and embedding. We abstract the details of such a Container so as to avoid restricting our model's commitment to any one approach. The Container we propose can only be modified by the Transformation System and consists of a

Tag File and two main components, namely an Evidence Container Unit and a Proof of Action History that is used to store evidence provenance information. Each of these components contains an integrity hash that is stored within the Container. The Container also includes an integrity hash of the entire Container at the particular point in time, providing a form of layered hashing. It is important to note that we leave the sealing of Containers for our future work as it is not the focus of this dissertation and further research is required to determine the optimal manner of performing such a task.

The Container's ID, contained in the Tag File, may be quite lengthy [59], depending on the approach taken, and may not semantically represent the Container's contents or its purpose. In order to aid in this regard, the Tag File incorporates further identification information, namely a title and a description, to provide a means for investigators to easily refer to a Container and to more easily determine the contents of the Container. The Tag File may also embed the results of a secondary tool (or more) which is used for verification purposes and references to the source(s) from which it is derived/created — whether it is physical evidence in storage and/or Evidence Containers. It may also reference other collected evidence that provides important information regarding the evidence content and how it is used — such as user manuals and device documentation. In this manner the Tag file uses embedding and referencing only for verification information, source information and for providing additional information regarding evidence and its usage. Our reason for doing so is to ensure consistency among various Evidence Containers, their contents and how they are interpreted. Let us now continue with the Evidence Container Unit which is derived from the Evidence Unit discussed by Turner.

Figure 4: The Evidence Container Structure



We restrict an Evidence Container to having a single Evidence Container Unit, so that regardless of what is stored within the evidence content, whether it is a corpus of evidence or not, all Evidence Containers will have a uniform and consistent structure. In this manner all evidence and their associated information will also be stored consistently, making the interpretation of the Evidence Container and its contents easier. The Evidence Container Unit

contains evidence content and evidence-related metadata. The evidence content can be used for digital evidence, physical evidence or a corpus of evidence by embedding and referencing other Evidence Containers. Evidence-related metadata consists of attributes for the particular Container, allowing corpuses of evidence to store attributes consistently with other Evidence Containers. These attributes are explicitly defined in an attribute class repository. The use of ontologies has shown that such an approach is possible, however further research is required to determine the most optimal approach for doing so and that is not the focus of this dissertation. Using an attribute class repository allows us to incorporate this functionality without committing to any one approach.

The repository focuses on storing attribute classes with their associated properties, including well-defined names, descriptions and a domain of possible values. The evidence related metadata consists of instances of these attributes that contain values matching those of the evidence content. An attribute instance, referred to simply as an attribute, is represented by a and the set of all attribute instances, referred to simply as attributes, is represented by the set $\mathbb{A}$. Each Container, x, then contains a subset of attributes (including an admissibility attribute), represented by $\mathbb{A}_x$, such that $\mathbb{A}_x \subseteq \mathbb{A}$. Using a repository for all attribute classes helps to ensure that attributes are explicitly defined and used consistently through all Containers (and all investigations). It also helps to overcome problems with semantic and syntactic heterogeneity. In this manner it will greatly aid the interpretation of an investigator's findings when they capture not only the evidence that serves a particular role in their reasoning, but also the attributes and the criteria that allow them to do so. Should evidence no longer be admissible, an investigator can use the Transformation System to modify the admissibility attribute, allowing these Evidence Containers to reflect the dynamic aspects of the investigation. An investigator can also use the Transformation System to add additional attributes and information regarding the Evidence Container. Any such changes made by the investigator are recorded within the Proof of Action History of the Container, which is continuously appended throughout the investigation.

The Proof of Action History, which we discussed previously, is used for capturing detailed Proof of Actions regarding digital and non-digital provenance events for the Container and its associated evidence content. In this manner, the Proof of Action History contains a more thorough Chain of Custody for a particular piece of evidence, allowing an investigator to support the reliability of the evidence. Let us now explore the transformations that are used for creating these Containers and adding the associated Proof of Actions. Before we proceed, it is important to note that in Chapter 5 we use a similar Container to capture an investigator's findings and their associated reasoning. By doing so, Finding Containers can reap the same benefits as the Evidence Container and the software components used to access and produce such Containers can be reused.

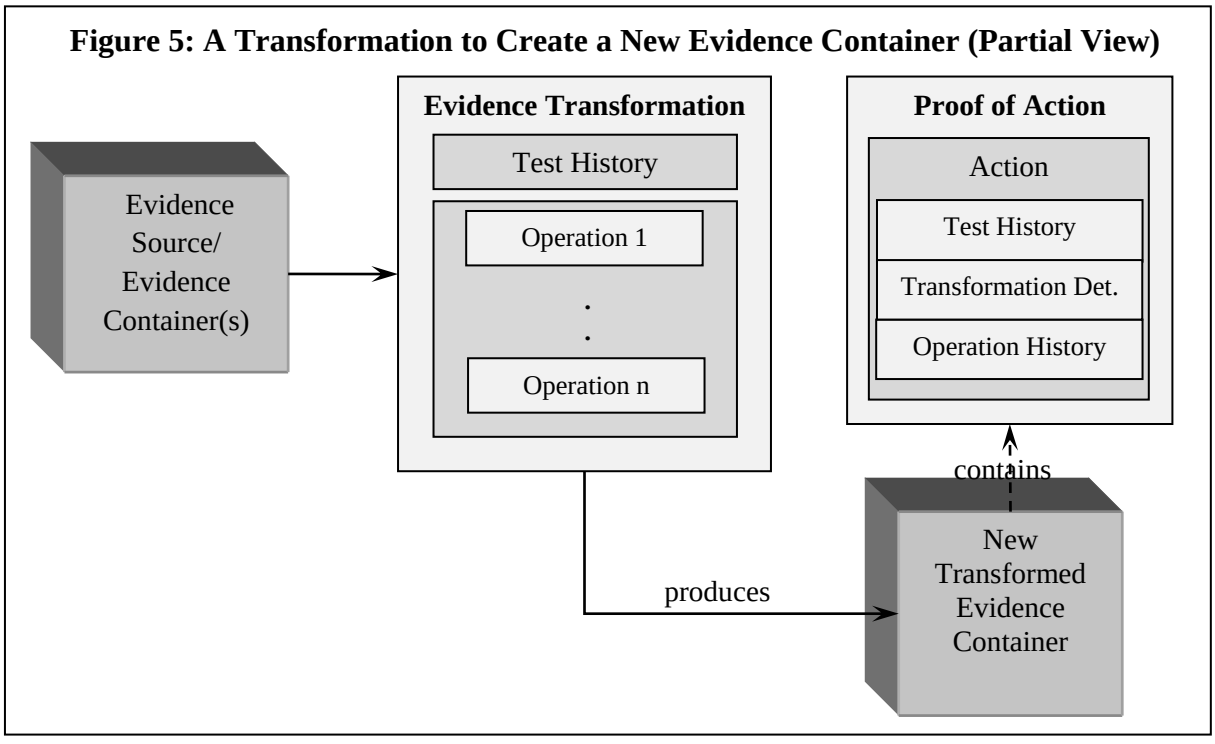
4.3.2 Evidence Transformations

A Transformation is used for five main reasons, namely to create a particular Container, to transform a Container, to append additional provenance information, to provide verification as a secondary tool, and to view a Container and its contents in a useful format. Each

Transformation is responsible for performing its necessary tests and for ensuring that it is calibrated correctly and functioning correctly. The Transformation Manager is then responsible for scheduling such periodic tests. These tests and their results are stored within the test history of the Transformation, allowing one to easily query the reliability of a particular Transformation.

Each Transformation contains the sequence of operations that are used for performing is particular task/transformation. Whenever a Transformation executes and transforms a Container, and/or produces a new one, the Transformation creates a Proof of Action. The Proof of Action contains the history of the operations performed, Transformation details — including the Transformation signature, preconditions and postconditions (as well as their results) — and the test history, as well as the investigator's rationale for performing the task/transformation. In the case of a viewing Transformation (namely a Transformation for viewing the Evidence Container and/or its contents) the viewing Transformation simply appends the Proof of Action appropriately to the source Container. Should a new Container be produced, two versions of the Proof of Action are created, one for the source Container(s) and one for the newly produced Container. The Proof of Action for the source Container(s) incorporates information regarding the newly created Container and a reference to it. The Proof of Action for the newly created Container contains information regarding the source Container(s) and a reference to it(/them). In this manner each of the Containers can be viewed in isolation to determine their entire lifecycle in the investigation and hence how they came to be in their current form.

Figure 5: A Transformation to Create a New Evidence Container (Partial View)



When a Transformation is used to create an Evidence Container for physical evidence, only a Proof of Action for the newly created Container is produced, containing a reference to the evidence in storage. Then when an investigator enters secondary physical evidence derived

from the physical evidence in such a Container (such as fingerprints), a Proof of Action is added to both these Containers (as was done previously) and the secondary Container also contains a reference to the secondary evidence (fingerprints) in storage. These Transformations allow investigators to enter various information about the evidence, which is then added to the Container, ensuring that all information regarding the evidence is available in a single location. Annotation Transformations are also available in the Transformation System and can be used to add additional provenance information through the use Proof of Actions, such as for non-digital events or for additional information regarding the imaging process. Whenever a Transformation creates a new Container, the Transformation is responsible for configuring the attributes of the Container using the attribute class repository. Transformations are also responsible for configuring the Tag File.

It is important to note that Transformations can reuse existing code, such as from existing Digital Forensic tools, and they do not have to be (re)coded from scratch. The financial and time investments that are put into creating existing Digital Forensic tools has been emphasized by Turner [68], and he discusses the importance of not simply disposing of these programs. Approaches such as the use of the Adapter design pattern [26][36] for Digital Forensic tools written using object-orientated code, allow one to adapt existing tools to conform to that of a Transformation. Using similar approaches allows one to reuse existing code as well as to adapt it, in order to add the required functionality of a Transformation. In order to help an investigator manage the various Transformations at their disposal and their periodic testing, we introduce the Transformation Manager.

4.3.3 Transformation Manager

There exists a mass of tools that an investigator has at their disposal [32] particularly due to the large number of growing Digital Evidence sources and the forms of analysis that can be performed on them. There may therefore be a large number of tools that an investigator can use for performing a particular task in an investigation. An investigator should choose tools based on whether they have the necessary functionality [24], and whether they produce results that are relevant, reliable, accurate and objective [10][22]. Unfortunately, testing is a complex and time consuming task, and the documenting of tests to support the reliability of the tool is an administrative burden (section 4.2.1). In order to aid the investigator in this regard, as well as to help them to find the correct Transformations and to be able to make more informed decisions regarding which Transformations they choose, we introduce the Transformation Manager.

The Transformation Manager is a facility that maintains a repository for all registered Transformations. It ensures that Transformations are functioning correctly — namely through various periodic tests — and it focuses on making these Transformations accessible to the investigator. The Transformation Manager aids investigators by making sure that all possible Transformations available for a particular task are displayed to the investigator, along with their test history (providing a form of reliability history) and the particular operations they perform. Investigators can then use this information to make an informed decision about which particular Transformation to use, as well as which Transformation to use for

verification purposes, should they wish to do so. The Transformation Manager also manages the updates of Transformations and ensures that updates are working correctly before they are committed. In addition, the Transformation Manager is responsible for ensuring that the Transformations do not conflict with one another.

Digital Forensic tools can interfere, or otherwise conflict with one another, as stated by NIST [32]. They explain how resolving and determining such conflicts can be difficult and time consuming. The test history, namely the calibration tests, can help in this regard and may provide the necessary information to convey when such conflicts occur, such as by demonstrating tool anomalies. The investigator can then use this information to aid them in isolate and determine conflicting tools (and processes), However, this is a manual and time consuming process. NIST explores a different approach and they explain how one may use tool segregation, to avoid such problems, namely by creating and running conflicting tools in separate virtual environments [32]. The Transformation Manager, together with the test and calibration process can help to set up and manage virtual environments that are well-suited for each tool. The benefit of using such an approach is that an investigator can use a single system to run multiple environments, each well-suited for the Transformations they require. The Transformation Manager also uses a registration and deregistration service to ensure that it is able to keep up to date with the latest developments in digital devices and media.

An important aspect when designing a system is to ensure that the system is not only able to take into account current problems and concerns, but that it can cater for future ones as well [7][26][36][60]. Earlier in our discussion on the Forensic Integration Architecture (FIA), we discussed how the FIA allows for the registration of interfaces and interpreters to ensure that it is able to keep up with future types of evidence. The Transformation Manager incorporates similar registration facilities for new Transformations, to allow the Transformation Manager to keep up with new types of evidence, new formats and new forms of analysis. In addition, the Transformation Manager is responsible for ensuring that these Transformations are up to date, well-tested and calibrated. Furthermore, the Transformation Manager can also be expanded to incorporate and test the credibility of an investigator before providing the investigator with access to the particular Transformation. The incorporation of such a credibility check may help to further the reliability of a Transformation's results, according to the Daubert Method. The Transformation Manager will however require detailed information about the investigator, such as their rank, skills, experience, training and publications, as well as their skills, experience and training with regards to using the particular Transformation.

4.4 Chapter Summary

In this chapter we introduced a particular Container, called the Evidence Container, which is derived from the DEB and SDEB and allows all evidence — digital and physical — to be used and treated uniformly. The Evidence Container provides a single location for encapsulating provenance information, evidence metadata and the evidence itself, allowing all information regarding evidence to be more easily accessible. Provenance information is represented using a Proof of Action History, allowing not only digital events regarding an

evidence's provenance, but also physical events to be recorded. The use of Proof of Actions helps ensure non-repudiation for these events and that these events are accurately timestamped, allowing for a more reliable timeline of events.

Transformations are responsible for creating, transforming and configuring such Containers and ensuring that they record detailed information regarding their actions and their reliability within the Proof of Actions. The Transformation Manager is then responsible for managing these Transformations and ensuring that they are working correctly before they are made available to the investigator. The Transformation Manager also incorporates registration capabilities to enable investigators to keep up with the advances in digital devices and media, and new Transformations.

In the next chapter we discuss the Finding Container and the Chain of Findings. The components discussed in Chapter 4 play an important role in the Finding Containers and hence our Chain of Findings. The incorporation of these components allows the Chain of Findings to meet a number of the requirements we specified in our problem statement. The requirements that are met by these components are repeated for ease of reference. *The Chain of Findings should incorporate a means for investigators to reliably transform their building blocks, as is needed, and to record a detailed audit trail of the transformation and its reliability. The Chain of Findings should also allow for the capturing of provenance information regarding not only the transformations performed but also physical events, and it should ensure that non-repudiation is a core aspect of all recorded events. The Chain of Findings should be designed with flexibility in mind to allow it to grow and adapt with the advancements in digital devices and media.* In the next chapter we investigate how we designed the Chain of Findings to ensure that it meets the remaining requirements that are specified in our problem statement.

Chapter 5

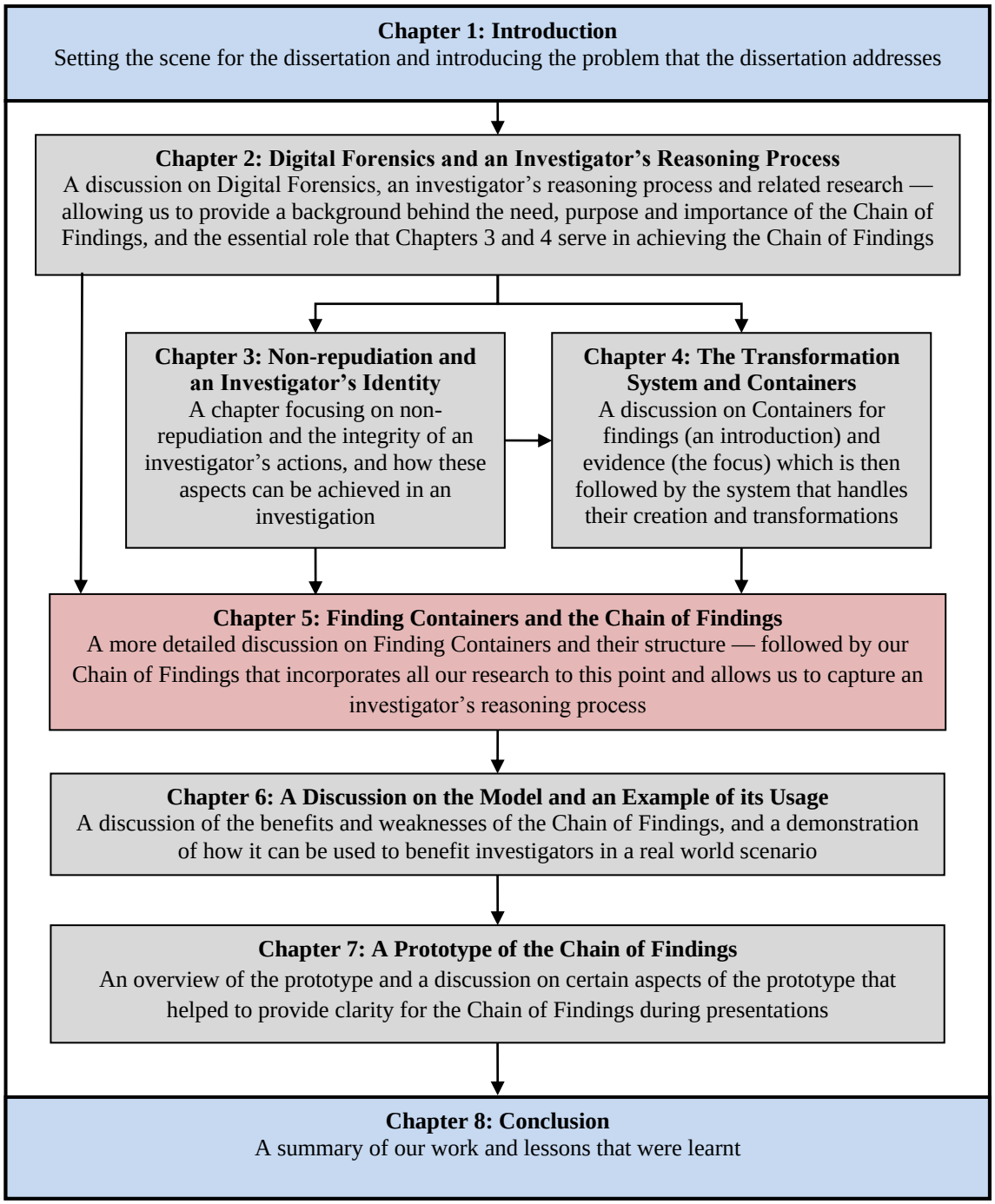
The Chain of Findings

Objectives

- To discuss the formal model used to capture an investigator's reasoning process that was followed during an investigation, in order to reason about and formulate arguments for their finding(s) and opinion(s) and to arrive at conclusions.
- To provide the necessary flexibility to allow an investigator to efficiently handle different incidents (eg. internal and external incidents)

The real voyage of discovery consists not in seeking new lands but seeing with new eyes
Marcel Proust

A Map of our Current Location in the Dissertation



Chapter 5: Finding Containers and the Chain of Findings

5.1 Introduction

Previously we discussed how an investigator's reasoning should be based on and supported by evidence (Cohen [15]), both digital and physical ([2][10][12]). We discussed how investigators must capture all information about their hypotheses during the investigation, the logical arguments behind them and the evidence which supports them (Rowlingson [58]). We also discussed how, during an investigation, an investigator must take into account inculpatory evidence, exculpatory evidence (Ahmad [2]) and evidence of tampering (Carrier [10]) into their reasoning, to ensure that their arguments are objective and are (more) sound. An investigator should furthermore take into account evidence in its entirety rather than in isolation, in order to construct a more complete picture of the incident and to draw more sound conclusions. Investigators should therefore not only rely on single sources of evidence (The National Institute of Justice [47]).

By taking into account evidence in its entirety, investigators can have a better understanding from which to form their reasoning, and to form more sound conclusions. We also discussed how an investigator should create theories that explain the facts, how an investigator should question their assumptions and how an investigator should not only explore alternate explanations, but also explain why they cannot hold (Casey [13]). An investigator should use multiple confirmations when doing so, in order to circumvent legal outcomes which may be inconsistent with logical reasoning (Cohen [15]). Another particular aspect that was emphasized was the importance of a detailed audit trail.

During an investigation an investigator will have to take a large number of steps in order to get evidence into its intended form such that the investigator can formulate their reasoning. Large volumes of data and other physical evidence may need to be collected during an investigation [11][68]. Investigators must take care to ensure that the integrity of evidence is preserved throughout the investigation, and hence evidence can be relied upon. Detailed audit trails are recommended in order to demonstrate the reliability of evidence throughout the investigation, from when it is collected to when it is presented, showing the integrity of the evidence is preserved [4]. Investigators should also capture a detailed audit trail regarding their conclusions and arguments in order to demonstrate their objectivity [47] during their reasoning process. In addition, investigators should factor in the certainty of their evidence and the arguments drawn from them, in order to be better prepared against challenges which may be presented against their findings (Casey [13]). Findings incorporate a large amount of information, time, and steps in order for the investigator to formulate and show, objectively, that the finding holds. Encapsulating all this information in a single location helps to facilitate the reconstruction of a finding and how it came to be, benefiting the presentation phase.

Evidence Containers (Chapter 4) provide a means to encapsulate all information regarding a particular piece of evidence. This information includes its attributes, the Proof of Action History (containing digital and physical provenance information), and the evidence itself. The attributes capture various characteristics about the evidence. The Proof of Action History captures all information about the Transformations performed, who performed them and the time that they were performed. The use of such a Container helps to ensure that all information regarding a particular piece of evidence can easily be retrieved from a single location, including how it came to be. Using a similar approach to reap such benefits and uniformity, we propose a Finding Container to capture and encapsulate all information regarding an investigator's finding and how they came to be throughout the investigation.

Finding Containers (overview provided in section 5.2) provide a means for investigators to capture and encode the reasoning behind their findings, using a well-defined structure. These Finding Containers play an important role in the Chain of Findings, and help ensure that the Chain of Findings meets the remaining requirements specified in our problem statement. They allow the investigator to capture their hypotheses, their arguments for why the hypotheses should be supported and refuted, the certainty with which they can make such claims, the evidence that they use to formulate such arguments and the conclusions they draw (the findings). The supporting and the refuting arguments allow investigators to demonstrate that they reasoned objectively, explored alternate explanations and that they factored in inculpatory evidence, exculpatory evidence and evidence of tampering into their reasoning and the conclusions they draw. These arguments are composed of the various confirmations for why they should hold, and the certainty they contribute, allowing investigators to factor certainty into their reasoning.

The National Institute of Justice discusses how the investigator's report must contain all supporting materials for their findings — including audit trails, testing documentation, the Chains of Evidence and the evidence itself. By using Evidence Containers as the foundation for these confirmations, investigators can capture the evidence that plays a role in the confirmation, as well as all information that supports the evidence. Existing findings may, however, also play a role in an investigator's arguments, and may also be used to derive further findings in the investigation, in a similar manner to Expert Systems (Chapter 2) and in logical reasoning [25][50]. By allowing existing Finding Containers to also be used as the foundation for these confirmations, an investigator can capture not only how evidence is used in their reasoning but also how existing findings are used as well. Overall, Finding Containers provide an investigator with a dynamic means for recording and encoding their reasoning throughout the investigation, the impacts that Containers have upon their reasoning, how these Containers are factored into their reasoning and finally how an investigator arrived at their conclusions.

In Chapter 2 we discussed the examiner's report and how investigators must capture their findings, the supporting evidence, and the steps taken by the investigator but also their identity and signature for non-repudiation purposes. The inclusion of a Proof of Action History (Chapter 4) for each Finding Container allows the Container to capture an investigator's steps during an investigation in more detail, providing non-repudiation and

ensuring a more thorough reconstruction is possible. In this manner, Finding Containers allow an investigator to capture the building blocks for their reasoning and the way in which they formulated their reasoning — and to then maintain an integrity hash over this information. Once we have completed our discussion on the Finding Containers, we then propose the Chain of Findings.

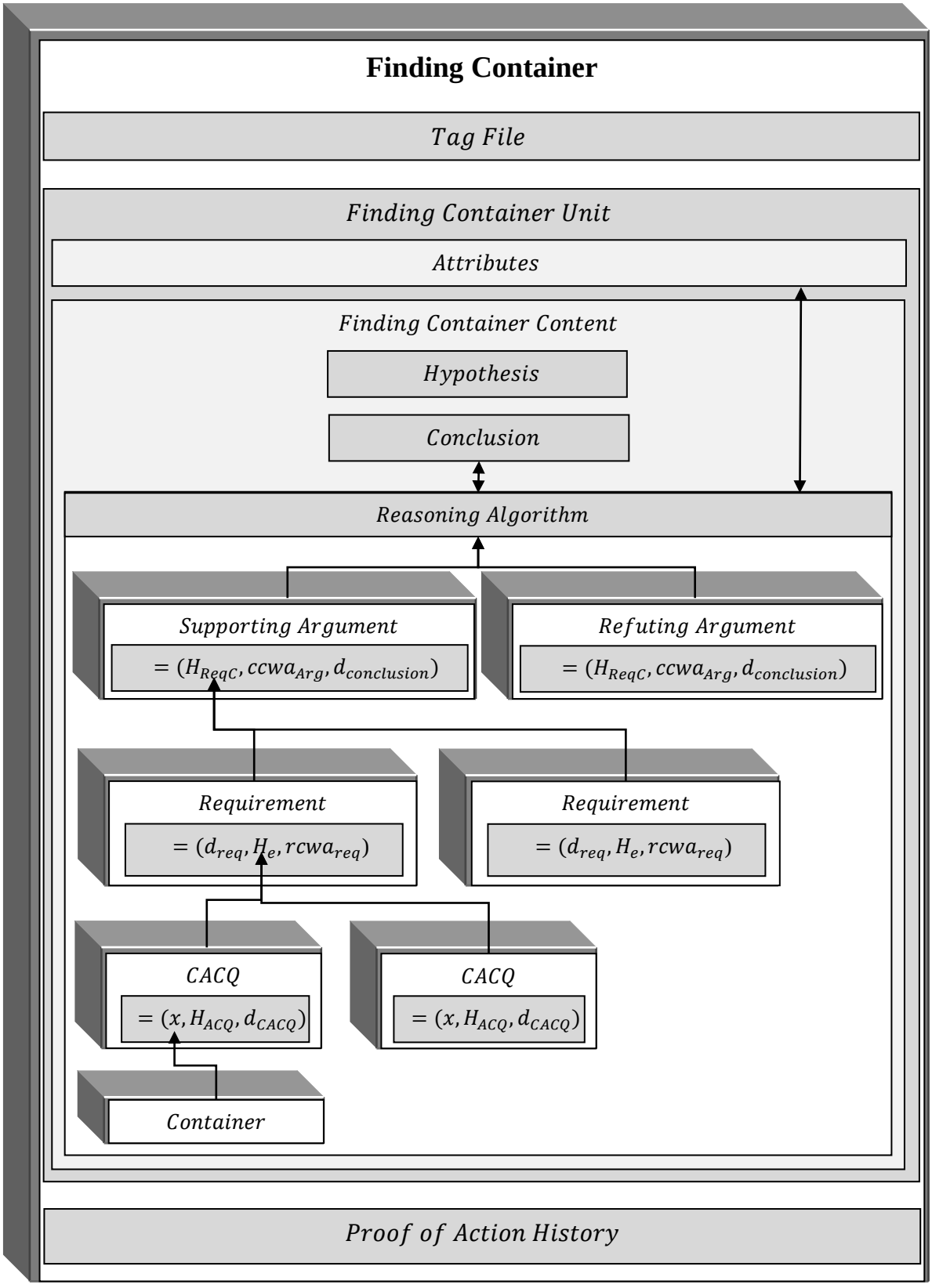
The Chain of Findings (section 5.7) is a model which incorporates all the research up to this point, from Time Stamping Authorities, Evidence Containers and the Transformation System to Finding Containers and is intended to meet the requirements specified in our problem statement. The Chain of Findings provides a means for investigators to capture and encode the reasoning process behind their findings. In Chapter 3 we discussed how the Chain of Custody provides a means for an investigator to capture the chain of parties who had custody of a particular piece of evidence during the investigation. The Chain of Findings provides the means for an investigator to capture their reasoning process — the relationships between Containers (Evidence and Findings), how an investigator uses them to create and structure their arguments, as well as the steps they followed. In this manner the Chain of Findings provides the means for an investigator to capture the chains of reasoning that was used in order for an investigator to arrive at the conclusions (findings) they did.

5.2 An Overview of a Finding Container

A Finding Container's structure (represented in Figure 6) that we propose resembles that of an Evidence Container (Chapter 4), containing a Tag file, a Container Unit, and a Proof of Action History. The Tag File and the Proof of Action History are the same as those of an Evidence Container. The Finding Container Unit is composed of attributes and the Finding Container Content. The attributes are similar to those used in Evidence Containers and they are used to record finding-related metadata. We include two particular attributes for Finding Containers, namely a supported attribute and a refuted attribute, which are used to store whether the hypothesis is sufficiently supported, refuted or neither. By storing whether the hypothesis is sufficiently supported, refuted or neither, this information is easily accessible to those who are interested in it, and those who want to use the Finding Container as a building block for further Finding Containers.

The Finding Container Content is where the biggest difference from that of an Evidence Container lies. Before we discuss its structure and its components, called Finding Components, we first give a broad overview of these Finding Components, beginning with the hypothesis. The hypothesis is the particular assumption that the investigator would like to support or refute, and it represents the main goal or purpose of the Finding Container. The conclusion represents the investigator's finding with regards to the hypothesis and is based on the investigator's reasoning, their Supporting Argument and their Refuting Argument (demonstrated by the arrows in Figure 6).

Figure 6: An Overview of a Finding Container



The Supporting Argument and the Refuting Argument allow an investigator to capture arguments for why their hypothesis should be supported and why it should be refuted, and the conclusions that should be drawn, should each of these Supporting and Refuting Arguments hold. We incorporate a conclusion into both of these Arguments, so that an investigator can encode the particular finding that they would make should the hypothesis be supported or refuted. The benefit of pursuing such an approach is that investigations may draw over large periods of time and, should more evidence be discovered at a later date, and the particular investigator no longer be available, the investigator's reasoning can be easily continued.

The investigator then uses these Arguments in order to factor in alternate explanations to reason which conclusion should hold, the supporting conclusion (the conclusion in the Supporting Argument) or the refuting conclusion (the conclusion in the Refuting Argument). In order to do so, each of these Arguments is composed of potential confirmations, with a particular certainty weight that describes why the particular Argument's conclusion should hold with the given certainty. To allow the investigator to capture when they are satisfied that an Argument is sufficiently confirmed and has a sufficient certainty, by its confirmations, we incorporate certainty thresholds into these Arguments. Should both the Supporting and the Refuting Arguments be sufficiently confirmed, the investigator will then have to reason about which particular argument should take priority over the other in this conflict. In order to capture and encode this form of reasoning the model introduces a Reasoning Algorithm.

The Reasoning Algorithm is used to evaluate the Arguments dynamically and to allow an investigator to capture how the particular conflict should be handled, should it occur. Note the arrows flowing from the Arguments to the Reasoning Algorithm in Figure 6. The Reasoning Algorithm evaluates these Arguments by confirming that their potential confirmations hold and that these Arguments meet their specified certainty thresholds. Should only the Supporting Argument be sufficiently supported and meet its certainty threshold, then the conclusion for the Finding Container is set to the supporting conclusion, the supported attribute set to true and the refuted attribute set to false. Similarly, should only the Refuting Argument meet its certainty threshold, then the conclusion is set to the refuting conclusion and the attributes are set accordingly. In the scenario that neither of the arguments meet their certainty thresholds, the conclusion for the Finding Container is then set to an empty string (ϵ) and both the supported and the refuted attribute are set to false. Should both the Supporting and the Refuting Argument be sufficiently confirmed, then the Reasoning Algorithm chosen by the investigator bases its decision on its given information to determine which Argument should be favoured, and which conclusion should be drawn.

In logic, arguments are composed of premises, and a conclusion that follows from the premises [25]. When one formulates such arguments one must take the necessary measures to prove that that the premises hold. One must also demonstrate that the conclusion does indeed follow from the premises, in order to circumvent criticism against one's reasoning [50]. These premises can be seen as a form of requirements for the conclusion where, should the requirements be met, then one states that they serve as a confirmation for the conclusion, and hence the conclusion can be derived. We build upon such an idea in our Supporting and

Refuting Arguments, and their associated confirmations, and allow investigators to represent these confirmations as a combination of Requirements.

These combination of Requirements represents the combination of conditions (or premises) that must be met to provide confirmation for the Argument's conclusion. The investigator is given the flexibility to combine the Requirements as they see fit, using brackets and logical connectives — allowing them to better capture and formulate the combination of conditions that will serve as a confirmation. The incorporation of such confirmations and a well-defined structure for them, ensures that investigators capture in detail what conditions have to be met in order for a particular Argument's conclusion to hold — allowing the investigator to capture deeper knowledge behind their reasoning.

In the Finding Container that we propose, the investigator then uses Containers, Findings and Evidence based, to support these Requirements and to show why they are met — encouraging the investigator to factor in multiple sources in the foundations for their Arguments. By capturing the Containers that support these Requirements, investigators can inherently capture the role that these Containers play in an investigator's reasoning, in more detail. The capturing of the combination of Requirements is also beneficial as it allows investigators to capture more in-depth information about the relationships between the Containers.

Containers may, however, be thrown out of court and no longer be admissible as the investigation progresses, requiring an investigator to possibly reformulate and recapture their arguments which rely on these Containers in their reasoning process. These reasons include privacy legislation, company policies [75], the judge's ruling [15] and the manner in which evidence was collected (particularly if sufficient precautions were not taken when performing the collections [4][73]). Containers may also have a large number of attributes, and the reasoning behind how and why the Container supports the Requirement (and hence plays a role in a particular Argument) may be lost if more in-depth information is not captured.

For example, let us assume there is a Requirement stating that "Frank Furt's account was logged into at 10:00 AM (UTC) on 14 January 2013". The investigator may support such a Requirement through the lengthy log record of the particular day, but why is this particular log record, containing a large number of entries, used to support the Requirement? This information should be more easily accessible and recorded in a structured manner because, as the investigation progresses, more and more pieces of evidence may be discovered and serve roles in the investigator's reasoning. Should this information not be captured, and the investigator no longer be available or be able to recall the logic behind their reasoning, this information might be lost. In order to cater for this we allow the investigator to capture the particular characteristic(s), namely the attribute(s), that allow the Container to support the Requirement, and the criteria they comply with which allows them to do so. The investigator can therefore capture how the Container supports the Requirement in greater detail. Continuing with our example, the investigator may then capture the *lines* attribute of the log record, and use the particular line number(s) of the log record (as a form criteria) which shows that Frank Furt's account was logged into at the particular time of interest. In this

manner the reasoning behind why the log record supports the Requirement is captured in greater detail, making it more easily accessible to those who desire such information.

The capturing of the Container, its attributes and their criteria has a number of other benefits. By using criteria, an investigator can also capture why a particular Container is not able to support a particular Requirement, such as CCTV footage that did not take place during the period of the incident. Investigations may continue over long periods of time and the importance of direction has been discussed in Chapter 2. Using the Transformation System to create empty Evidence Containers, referred to as Null Containers, can aid in this regard. The investigator can then use the identification information contained in the Tag File, such as the title and the description, to help emphasize what evidence is needed to help support a particular Requirement. The investigator can then use these Null Containers, along with the attributes and their criteria to create further guidelines on the values its attributes must comply with. This approach allows investigators to provide direction for themselves and fellow investigators on what evidence is needed and the criteria it must comply with, which provides greater flexibility for the investigator. Flexibility is an important requirement specified in our problem statement. Should the correct evidence be discovered, it can then be substituted with the Null Container or entered into the Null Container.

The use of criteria also provides a means for an investigator to capture when a Container can serve a particular role in their reasoning process. Such as an existing Finding Container, which must be supported (attribute *supported* = true) or an Evidence Container which must not be admissible (attribute *admissible* = false) in the investigation. In this manner the investigator can provide more in-depth information about not only which Containers support the Requirement, but also what particular characteristics of the Container (would) allow the Container to do so, and how.

Unfortunately an investigator will not always have the time and resources to continuously check whether these Containers meet their criteria or not. An investigator may therefore not always be able to determine the impacts that they have on the investigator's Requirements, and on the confirmations which incorporate these Requirements. In order to cater for this, the Requirements are composed of particular queries, which allow the investigator to capture the Containers, their attributes of interest and the criteria they must comply with to serve their particular role. These queries can then be dynamically executed. Casey [13] states that an investigator should take into account the certainty of their evidence and the certainty of conclusions they draw from this evidence (Chapter 2). To incorporate such certainty with regards to their evidence and the contribution they make, each particular query is given a certainty weight based on the contribution it makes to the Requirement being met. The Requirements are then also allocated a certainty threshold, to enable investigators to capture the certainty that the Requirement must obtain before the investigator is satisfied that the Requirement is met. Only if the queries, which are dynamically evaluated throughout the investigation, are successful and meet their specified criteria, is their certainty contributed towards that of the Requirement. The Requirement is then only able to serve its role in its associated confirmation, should it attain its specified certainty threshold.

We name the particular type of query that allows an investigator to capture the Container, its attributes, and its criteria that serve a role in a particular Requirement, a Container Attribute Checking Query ($CACQ$). These $CACQs$ are used to incorporate all the attribute information, the criteria they must comply with, and the rationale behind them in a well defined structure, allowing all information regarding the Container to be easily retrieved. A large number of attributes might be necessary for an investigator to capture their reasoning, and these attributes might each have a number of criteria, which may be included for different reasons. In order to capture this information within the $CACQ$, we propose smaller Attribute Checking Queries ($ACQs$) that each store information about a particular attribute of interest, its associated criteria and its rationale. The $CACQ$ then incorporates each of these smaller $ACQs$, allowing an investigator to view the query at the granularity that suits their needs, providing more clarity on what aspects of the Container are of interest and their purpose, should such information be needed.

Each of the Finding Components that we have discussed up to this point allows an investigator to capture information on how it achieves its role in the investigator's reasoning process. What of why they were used, why they contribute a particular certainty, and why a Container must meet the specified criteria? An investigator can, for example, use a separate document to capture such information or rely on their memory to recall such information when presenting their Findings. Investigations may however continue over large periods of time, and all information regarding a Finding should be encapsulated within the Finding, making such information more easily accessible. To allow the investigator to capture this information, and answer these types of questions, we incorporate a particular element into many of these Finding Components, which we call a description element. The investigator can then use these description elements to describe the rationale behind each of the Finding Components.

It is important to note that certainty in the Finding Container and its associated Finding Components is purely optional, and hence Finding Containers provide the flexibility for certainty to be incorporated only if the investigator wishes to do so. In this manner investigators can use the Finding Containers to capture the reasoning process they followed and in addition, they can incorporate certainty throughout their reasoning or only in the Finding Components and Containers where they are needed. The incorporation of certainty provides a means for these Finding Containers to be dynamically evaluated, and allows the Finding Container to determine which conclusions to draw, based on the reasoning encoded by the investigator. Overall the flexibility (one of the requirements set out in our problem statement) is beneficial as it allows investigators to choose an approach that best suits their needs, providing additional complexity and dynamic reasoning capabilities, only if the investigator wishes to pursue such an approach and utilise such behaviour.

We have elaborated on how each of the Finding Components are used with one another, providing an overview of the Finding Container. Now we discuss them in more detail. We use a bottom-up approach for doing so, so that we can explain the Finding Container from its smallest elements, and conclude with how they are used to formulate an investigator's

conclusions. Before we begin our discussion, we first define a number of positive integers (i, j, k, p, r, q, z), which are used frequently in the definitions:

$$i, j, k, p, r, q, z \in \mathbb{Z}^+, \text{ where } \mathbb{Z}^+ \text{ is the set of all positive integers}$$

While we use $\mathbb{R}$ to represent an investigator's possible certainty, we constrain all elements of the set in this dissertation such that

$$\forall n \in \mathbb{R}, n \geq 0$$

We incorporate zero for cases where the investigator chooses not to incorporate their certainty into their reasoning. By using real numbers, an investigator is given the freedom to specify positive and negative certainty. In this chapter and in our dissertation we do not investigate negative certainty, leaving such matters for future research. We, however, use $\mathbb{R}$ to avoid unnecessarily constraining the model at this point.

We also use the set of all strings, which we denote with $\mathbb{S}$, when defining Finding Components, where these strings are any combinations of alphanumeric characters, punctuation, mathematical symbols and other symbols which may be needed by the investigator to capture their descriptions. We now examine these Finding Components in more detail using a bottom-up approach, beginning with the $CACQ$.

5.3 A Container Attribute Checking Query (C*ACQ*)

A $CACQ$ allows an investigator to capture the Container (through the use of its ID), the particular attributes of interest and the criteria that they (must) comply with in the form of a query, which we define as a tuple of the form (x, H_{ACQ}, d_{CACQ}). In the tuple, x represents the particular Container that is of interest to the investigator and that the $CACQ$ must query. The second element of the tuple, H_{ACQ}, is a set used to hold all $ACQs$, which each capture attributes of the Container and the criteria that these attributes must comply with. We refer to such a set as a holder set, as it *holds* these smaller $ACQs$ and we therefore represent the set with a H. The $ACQs$ contained in H_{ACQ} allow an investigator to capture how the Container serves its particular role in the investigator's reasoning. The third element of the tuple is a detailed description of the $CACQ$ and it is represented by d_{CACQ}.

The description element allows an investigator to capture their explanation of why the Container meeting its specified criteria, (specified by the smaller $ACQs$), plays a role in their Finding. This provides the investigator with the capability, as the investigation progresses, or in the post investigation phases, to determine the $CACQs'$ purposes, and to more easily understand the foundation from which their Findings were created. Overall, the $CACQ$ allows one to place all the Container information, regarding the *how* (the various $ACQs$) and *why* into a single non-volatile object. Before we elaborate further on a $CACQ$ and define its elements, let us first define an ACQ and discuss how it performs its role in the $CACQ$.

5.3.1 The Formal Definition of an ACQ

An Attribute Checking Query (ACQ) allows an investigator to capture information about an attribute and the particular criteria that the attribute must comply with. It also allows an investigator to describe why the attribute and its specified criteria is of interest. We define an ACQ as a tuple of the form $(a, \mathcal{C}_s, d_{ACQ})$, where a is the attribute of interest, $\mathcal{C}_s$ is the particular set of criteria and d_{ACQ} is the investigator's description of the ACQ. In order to formally define an ACQ, we first define the set of all $ACQs$ as S_{ACQ} and we then define an ACQ, denoted ACQ_i, as follows:

$$\forall ACQ_i \in S_{ACQ}\ , ACQ_i = (a, \mathcal{C}_s, d_{ACQ}), \text{ such that}$$

$$a \in \mathbb{A}, \text{ where } \mathbb{A} \text{ is the set of all attributes (Chapter 4)}$$

$$\mathcal{C}_s \subseteq \mathcal{C}_a, \text{ where } \mathcal{C}_a \text{ represents the set of all criteria that is applicable to a particular attribute, } a$$

$$d_{ACQ} \in \mathbb{S} \text{ where } \mathbb{S} \text{ is the set of all strings and } d_{ACQ} \text{ is used to capture the description and the rationale of } ACQ_i.$$

The set S_{ACQ} contains all possible $ACQs$ and, by grouping these $ACQs$ into a $CACQ$ that focuses on a particular Container x, an investigator can capture the attributes and criteria that a Container must comply with to serve its role in the investigator's reasoning. It is important to note that such an approach allows an investigator to specify $ACQs$ regarding attributes which do not apply to the Container. The benefit of such an approach is that an investigator can specify guidelines for an investigation using Null Containers, and they can use these $ACQs$ to demonstrate why a Container is not able to perform a particular role in an investigator's reasoning.

Unfortunately, there exists a large number of attributes that may be used to configure any Container, particularly due to the large number of different types of potential evidence. Searching for the relevant attributes to construct a ACQ can be time-consuming, especially if an investigator would like to specify a ACQ for an attribute a Container already has. The model therefore allows an investigator to only use attributes that are already contained within the Container (represented by $\mathbb{A}_x$), reducing the number of attributes that an investigator must navigate through in order to form their ACQ. By doing so we can refine the set of $ACQs$ to only those that are applicable to a particular Container x, forming a subset S_{ACQ_x} which can be defined as follows:

$$\forall x \in \mathbb{C}\ , \text{ where } \mathbb{C} \text{ is the set of all Containers}$$

$$\exists S_{ACQ_x} \subseteq S_{ACQ} \text{ such that}$$

$$\forall ACQ_i \in S_{ACQ_x}\ , \text{ where } ACQ_i = (a, \mathcal{C}_s, d_{ACQ}), \text{ then}$$

$$a \in \mathbb{A}_x, \text{ where } \mathbb{A}_x \subseteq \mathbb{A} \text{ and } \mathbb{A}_x \text{ is the set of all attributes contained in } x \text{ and}$$

$$\mathcal{C}_s \subseteq \mathcal{C}_a \text{ and}$$

$$d_{ACQ} \in \mathbb{S}.$$

Now that we have formally defined an ACQ, let us now define a $CACQ$.

5.3.2 The Formal Definition of a $CACQ$

In order to formally define a $CACQ$, we first define the set of all $CACQ$ objects as S_{CACQ} and then we define a $CACQ$, denoted $CACQ_i$, as follows:

$$\forall CACQ_i \in S_{CACQ}, CACQ_i = (x, H_{ACQ}, d_{CACQ}) \text{ where}$$

$$x \in \mathbb{C}, \text{ and } x \text{ is the Container of interest}$$

$H_{ACQ} \subseteq S_{ACQ}$, and H_{ACQ} represents the holder set which contains all the $ACQs$ which together capture the list of attributes and criteria that the Container x should comply with

$$\forall\, ACQ_j = \left(a_j, \mathcal{C}_{s_j}, d_{ACQ_j}\right), ACQ_k = \left(a_k, \mathcal{C}_{s_k}, d_{ACQ_k}\right) \in H_{ACQ} \text{ such that}$$

$$ACQ_j \neq ACQ_k \text{ then}$$

$$a_j \neq a_k \text{ because if } a_j = a_k \text{ then}$$

$$\exists ACQ_p = \left(a_p, \mathcal{C}_{s_p}, d_{ACQ_p}\right) \in S_{ACQ} \text{ such that}$$

$a_p = a_j = a_k$ and $\mathcal{C}_{s_p} = \mathcal{C}_{s_k} \cup \mathcal{C}_{s_j}$ and d_{ACQ_p} incorporates the information from both d_{ACQ_j} and d_{ACQ_k} — hence

ACQ_j and ACQ_k could be replaced by ACQ_p

$d_{CACQ} \in \mathbb{S}$, and d_{CACQ} is used to capture the description and rationale of $CACQ_i$.

It is important to note that in the formal definition of a $CACQ$, we state that the holder set (H_{ACQ}) may only contain a single ACQ for each distinct attribute in the Container. We demonstrate the purpose of this restriction by arguing that, should there exist two $ACQs$ in a $CACQ$ regarding the same attribute, they could be replaced by a single ACQ (ACQ_p) which incorporated both the criteria from the individual $ACQs$. A $CACQ$ may consist of a number of smaller $ACQs$ and, by incorporating such a restriction, all information regarding a particular attribute (namely the criteria it must comply with and the rationale behind it) can be contained in a single ACQ, instead of being distributed over multiple $ACQs$. In this manner the reasoning process that is captured by the $CACQ$ can be clearer and more concise, and hence more easily understood. The restriction also helps to ensure that the model can effectively enforce further restrictions.

For each $CACQ$ we impose another two restrictions. These restrictions apply to Evidence Containers and Finding Containers. The restriction for $CACQs$ which apply to Evidence Containers, namely Containers of the set $\mathbb{C}_E$ (where $\mathbb{C}_E \subseteq \mathbb{C}$), is that the $CACQs$ must contain a particular ACQ that specifies whether the Container must be admissible or not in order for it to serve its purpose in the investigator's Finding. Should the investigator specify

that the particular Container should be inadmissible, hence ($admissible = false$), then the $CACQ$ may not contain any other $ACQs$. The main reason behind this is that, should an investigator require a particular Evidence Container to be inadmissible in order for it to serve its purpose in their reasoning, then the investigator cannot rely on the Container's other attributes in their reasoning. This particular restriction can be more formally specified as follows :

$\forall CACQ_i = (x, H_{ACQ}, d_{CACQ}) \in S_{CACQ}$ where $CACQ_i$ focuses on querying a particular Evidence Container, x, such that $x \in \mathbb{C}_E$ (the set of all Evidence Containers),

$\exists\, ACQ_j \in H_{ACQ}$ such that $ACQ_j = (admissible, \{admissible = b\}, d_{ACQ})$ where

$b \in \mathbb{B}$, and $\mathbb{B}$ is the Boolean set namely $\{True, False\}$

but

if b is $False$ then

$\nexists\, ACQ_k \in H_{ACQ}$ such that $ACQ_j \neq ACQ_k$ and hence $H_{ACQ} = \{ ACQ_j \}$

The investigator can use such inadmissibility criteria to construct counter arguments or to form backup arguments to factor in the dynamic nature of investigations. The investigator can do so by showing that, should a particular Container be inadmissible, how arguments should then be constructed to support the argument, or why a certain opposing argument can then not hold.

We also place a restriction on all the Finding Containers, relating to the supported and the refuted attributes. The restriction enforces that, for every $CACQ$ based on a Finding Container, either the Container must be supported to serve its role, the Container must be refuted to serve its role or the Container should be neither supported nor refuted in order to serve its role. This helps to ensure that investigators will specify the particular state that a Finding should be in, in order for it to serve its role in their reasoning. We can more formally specify the restriction as follows:

$\forall CACQ_i = (x, H_{ACQ}, d_{CACQ}) \in S_{CACQ}$ where $CACQ_i$ focuses on querying a particular Finding Container, x, such that $x \in \mathbb{C}_F$ (the set of all Finding Containers),

either $ACQ_j \in H_{ACQ}$, $ACQ_k \in H_{ACQ}$ or $ACQ_l, ACQ_p \in H_{ACQ}$ such that

$ACQ_j = (supported, \{supported = true\}, d_{ACQ})$ and hence the Finding Container must be supported to achieve its goal

$ACQ_k = (refuted, \{refuted = true\}, d_{ACQ})$ and hence the Finding Container must be refuted to achieve its goal and

$$ACQ_l = (supported, \{supported = false\}, d_{ACQ}) \text{ and}$$
$$ACQ_p = (refuted, \{refuted = false\}, d_{ACQ}) \text{ and hence the Finding Container cannot be supported or refuted.}$$

Now that we have discussed the $CACQ$ and the associated $ACQs$ in more detail, we must discuss how they are evaluated. In order to evaluate these $CACQ$s we discuss the evaluation function E_{CACQ} and the evaluation function E_{ACQ} which evaluates smaller atomic $ACQs$, from which the $CACQ$ is composed.

5.3.3. The Evaluation of an $CACQ$

In order to evaluate a $CACQ$ we introduce an evaluation function, namely E_{CACQ}. The evaluation function determines whether or not the Container specified in the $CACQ$ complies with its required criteria (contained in its $ACQs$) , for it to serve its role in the investigator's Finding. In order to do so, the E_{CACQ} takes a particular $CACQ$, namely $CACQ_i$ as a parameter, and produces a Boolean value that represents whether or not the Container meets its specified criteria. We define E_{CACQ} as follows:

$$E_{CACQ}(CACQ_i) \rightarrow \mathbb{B}, \text{ where } \mathbb{B} \text{ is the Boolean set namely } \{True, False\} \text{ and}$$

$$CACQ_i \in S_{CACQ}$$

In this manner E_{CACQ} confirms whether a Container meets all its specified criteria, allowing these evaluations to be automated throughout the investigation, however it relies on the evaluation of the $CACQ's$, smaller $ACQs$, in order to do so. In order to evaluate these $ACQs$, another evaluation function is introduced, namely E_{ACQ}. E_{ACQ} takes two parameters, the first being the particular Container x, and the second being a particular ACQ which must be compared against Container x. E_{ACQ} then performs the comparison by checking that the Container x does indeed have the attribute specified in the ACQ, and that the criteria specified in the ACQ complies with the value of the attribute of the Container. E_{ACQ} can therefore be represented as

$$E_{ACQ}(x, ACQ_i) \rightarrow \mathbb{B} \text{ , where}$$

$$x \in \mathbb{C} \text{ and}$$

$$ACQ_i \in S_{ACQ}.$$

The E_{CACQ} evaluates a $CACQ$ by first evaluating each of its $ACQs$ contained in its holder set (H_{ACQ}) and then logically ANDing ($\wedge$) them. In this manner E_{CACQ} can then confirm whether a Container, x, meets all its specified criteria. We represent the E_{CACQ} function as follows:

$$\forall CACQ_i = (x, H_{ACQ}, d_{CACQ}) \in S_{CACQ} \text{ where}$$

$$H_{ACQ} = \{ACQ_1, ACQ_2, ..., ACQ_n\} \text{ where } 1 \leq n, n \in \mathbb{Z}^+ \text{ then}$$

$$E_{CACQ}(CACQ_i) = E_{ACQ}(x, ACQ_1) \wedge E_{ACQ}(x, ACQ_2) \wedge \ldots \wedge E_{ACQ}(x, ACQ_n)$$

The use of the evaluation functions (E_{CACQ} and E_{ACQ}) and the declarative nature of the $ACQs$ and hence the $CACQs$, allows the investigator to specify what criteria must be confirmed against which Containers. These evaluation functions free the investigator from having to manually perform these evaluations and from specifying how these evaluations should be done. This separation of concerns allows the logic and functionality for the evaluation function to be encapsulated within them. By doing so, these evaluation functions can choose the most optimal approach for performing evaluations. The most optimal approach will depend on factors such as the execution environment, available resources and various other factors that may affect its efficiency.

Each of the $E_{ACQ}s$ involved in the E_{CACQ} can then choose the most optimal means for achieving its desired behaviour when evaluating the smaller ACQ objects, allowing for greater efficiency for the E_{CACQ}. An investigator may use a number of $CACQs$ and hence $ACQs$ when forming their Finding Container for an investigation and, by ensuring that each of the $CACQs$ are evaluated as efficiently as possible, it aids in improving the efficiency of the evaluation of the investigator's arguments as a whole. This is particularly so due to the fundamental role that $CACQs$ play in the Finding Container.

5.3.4. Summary of the $CACQ$

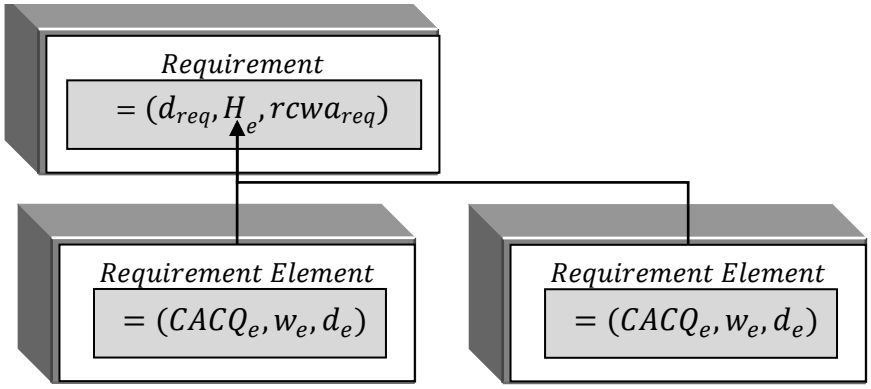
The intention of a $CACQ$ is to provide a single means to describe not only which Containers play a role in the investigator's arguments, but also how they do so and why. The *how* is achieved through the use of the $ACQs$, which allow an investigator to specify the attributes of the Container and the criteria that they must comply with in order to serve their role in the investigator's reasoning process. The *why* is captured through the various description elements incorporated into the $CACQs$ and $ACQs$. We now look at Requirements in a Finding Container and how $CACQs$ are used to support these Requirements.

5.4 A Requirement

A Requirement represents a particular condition that must be met in order for an investigator to provide a confirmation for why a hypothesis should be supported or refuted. By allowing investigators to record the $CACQs$ which support the Requirement they are able to capture how Containers support these conditions and, hence, provide more in-depth information on the role Containers play in an investigator's reasoning. These $CACQs$ may each support the Requirement and show that it holds with a particular certainty, where this certainty is based on factors such as the reliability of the Container and the support or proof it contributes to the Requirement. We would also like to allow investigators to capture why the particular $CACQ$ helps support the Requirement and why the investigator believes the $CACQ$ supports the Requirement with the associated certainty. In order to cater for this, and to allow investigators to capture this information in a well defined structure, we introduce a Finding Component, which we call a Requirement Element.

Each Requirement Element allows an investigator to capture a $CACQ$, the certainty weight the $CACQ$ contributes and a description behind why the $CACQ$ helps to support the particular Requirement with the associated certainty. All Requirement Elements which support a particular Requirement are then grouped into the Requirement, as is shown in Figure 7, allowing this information to be stored within the Requirement and to be easily located.

Figure 7: A Requirement and its Requirement Elements



Our intention for the Requirement is to provide a means for an investigator to capture what the particular Requirement is, the Requirement Elements that support why the Requirement should hold, and the certainty threshold for the Requirement. We also wanted to allow an investigator to capture how they would like to represent their certainty threshold and the certainty that each of the Requirement Elements contribute. Whether they would like to specify the number of required supporting Requirement Elements as a certainty threshold and each Requirement Element have a minimum certainty of one, or perhaps if they would like to use a percentage based approach where thresholds and certainty weights are represented as percentages. Should an investigator decide to incorporate certainty into their Requirements, we wanted to allow the investigator to always be able to receive dynamic feedback on their progress, whether they have reached their certainty threshold or not and how changes impact the Requirement's certainty.

We also wanted to provide investigators with the flexibility to not have to incorporate certainty into their Requirements, should they so wish, allowing them to focus instead on capturing their reasoning in a structured manner. We also wanted to capture how an investigator uses these certainty weights to determine whether the Requirement's threshold is met, and to encode this information so that it can be reused for other Requirements. The incorporation of such information is not only beneficial in the presentation phase, but also for those who want to understand and build upon another investigator's work. We incorporate such capabilities through the use of a new Finding Component which we call the Requirement Certainty Weighting Algorithm ($RCWA$).

Each $RCWA$ is composed of four main aspects, namely a domain of weights (D_{RCWA}), a domain of thresholds (T_{RCWA}), a particular certainty weighting algorithm (A_{RCWA}) and a description (d_{RCWA}). The domain of weights represents the possible weights that can be

assigned to Requirement Elements, based on the certainty they contribute to a Requirement being met. The domain of thresholds is used to represent the particular thresholds, t, such that $t \in T_{RCWA}$, which can be used to represent the certainty level/confidence level that an investigator would like the Requirement to attain before the investigator is satisfied that the Requirement is sufficiently supported. The certainty weighting algorithm, A_{RCWA}, is the particular algorithm that takes the Requirement Elements and their associated weights, as well as the threshold chosen by the investigator, t, and determines whether the Requirement is sufficiently supported.

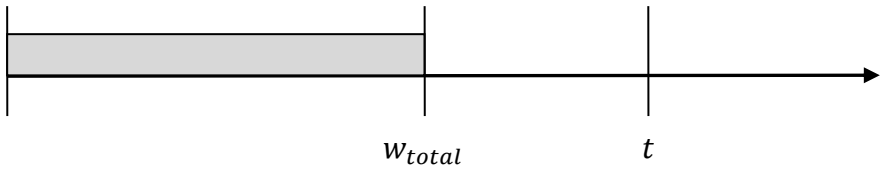
The certainty weighting algorithm (A_{RCWA}) is used to iterate through each of the Requirement Elements, namely those contained in H_e of a Requirement, and to confirm that their $CACQs$ hold. For each particular Requirement Element that evaluates to true and complies with the conditions of the A_{RCWA}, the certainty weighting algorithm then uses the Requirement Element's weight to calculate a certainty total for the Requirement. We represent the calculated certainty total as w_{total} in this dissertation. The certainty weighting algorithm then determines whether this total meets the required certainty level specified by the investigator, namely threshold t, to determine if the Requirement is sufficiently supported. In this manner all this information on how this is performed is contained within A_{RCWA}.

The final aspect of a $RCWA$ is the description (d_{RCWA}), which is used as a means to capture information about the $RCWA$, what its purpose is, and how its A_{RCWA} functions. The description helps the intended audience to understand what the certainty weighting algorithm is and how it works, as well as aiding investigators who are searching for a $RCWA$ to suit their needs. Should the investigator not be satisfied or find an existing $RCWA$ that meets their needs or determines a Requirement's certainty in the manner that they desire, they can simply create and add their own $RCWA$ to the model.

While the $RCWA$ captures the domain of weights, the domain of thresholds and the manner in which certainty is calculated for a Requirement and tested against the Requirement's certainty threshold, we would like to allow investigators to capture their chosen threshold, and record why they chose the threshold and the $RCWA$ they did. In order to do so we define an instance of a particular $RCWA$ (denoted $rcwa$) as a specialisation of a $RCWA$, which inherits the domain of weights (D_{RCWA}) and the certainty weighting algorithm (A_{RCWA}). The instance differs from the associated $RCWA$ in that, in place of T_{RCWA} (the domain of thresholds), it allows an investigator to capture the chosen threshold, $t \in T_{RCWA}$ and the instance is used to store the calculated certainty total w_{total} for a Requirement. By storing this information (along with the chosen threshold) in the same location, the model can more easily provide feedback on an investigator's progress in an investigation, such as that shown in Figure 8. The instance also includes a description element that is used to capture the rationale behind why the investigator chose a particular $RCWA$ for a Requirement, and the rationale behind them choosing the particular certainty threshold, t. By storing the instance containing all the certainty information regarding the Requirement within the Requirement, all such information can be more easily located and viewed when needed, helping to

incorporate such information within the Requirement without polluting the Requirement with complexity. Overall, these $RCWAs$ and their instances provide a means for investigators to factor in the certainty that they place in their Requirement Elements and in Requirements as well. However, an investigator may not always require such a means or like to use such an approach.

Figure 8: Progress Feedback using w_{total} and t



Where w_{total} is the calculated certainty total at a particular point in time with the use of a certainty weighting algorithm and t is the desired certainty level that an investigator would like to attain before a Requirement can be seen as sufficiently supported. The model then uses this information to provide the investigator with dynamic feedback with regards to their progress, throughout the duration of the investigation.

These $RCWAs$ and their associated instances have their benefits and their uses with regards to capturing certainty in an investigation, however in certain situations they can become a hindrance for the investigator, or simply may not be needed. In order to provide the investigator with the flexibility to choose whether or not to incorporate such behaviour, the model includes a particular $RCWA$, called the Null $RCWA$. The Null $RCWA$ has a domain of weights and a domain of thresholds which are both equal to the set $\{0\}$ — allowing an investigator to capture Requirements and Requirement Elements without concerning themselves about certainty and certainty weights. The model also provides investigators with the capability (or rather the flexibility) to use different $RCWAs$ for different Requirements, allowing the investigator to capture their reasoning in the manner that best suits their needs, and to switch between them as they see fit. The model therefore does not constrain an investigator to a particular approach, but instead allows them to choose the approach that best suits their needs. We discuss $RCWAs$ in more detail in section 5.4.2, where we also provide further examples of potential $RCWAs$ so that we can emphasize how they can be used. Now that we have discussed the various elements that make up a Requirement in the model, we now elaborate on how a Requirement is represented in the model.

A Requirement allows an investigator to capture what the Requirement is, the Requirement Elements which support the Requirement and all certainty information regarding the Requirement. In order to achieve this, we chose to represent a Requirement as a tuple of the form $(d_{req}, H_e, RCWA_{req})$, consisting of a description of the Requirement, a holder set H_e that contains all the Requirement Elements which support the Requirement and an instance of

a $RCWA$. The description element of a Requirement differs from previous description elements we have discussed up to this point, as the previous ones have mainly focused on capturing the *why* and are usually included at the end of the tuples. d_{req}, however, is at the beginning of the Requirement tuple and it focuses on answering *what* the particular Requirement is, and can be seen as the title of the Requirement. The holder set (H_e) consists of only Requirement Elements that support the Requirement, should they hold. $rcwa_{req}$ is an instance of a particular $RCWA$ that is chosen by the investigator to determine when a Requirement is sufficiently supported by its Requirement Elements.

In order to formally define a Requirement, we first define the set of all possible requirements for Finding Containers as S_{req}, and each particular requirement, req_i , is then an element of this set such that

$$req_i \in S_{req}$$

We then define the set of all Requirement Elements as S_e and then each particular Requirement Element, e_j, is then an element of this set such that

$$e_j \in S_e$$

We also define the set of all $RCWAs$ as S_{RCWA} and then each particular $RCWA$, $RCWA_k$, is then an element of the set such that

$$RCWA_k \in S_{RCWA}$$

We now define a requirement, req_i as follows

$$\forall req_i \in S_{req}\ , req_i = (d_{req}, H_e, rcwa_{req}) \text{ where}$$

$d_{req} \in \mathbb{S}$ and d_{req} describes *what* the requirement, req_i, is

$H_e \subseteq S_e$ such that $H_e = \{e \in S_e | \ e\ \mathcal{R}\ req_i\ \}$. H_e is therefore a relational set such that e is a Requirement Element and $\mathcal{R}$ is a particular supports relation, where if and only if e supports req_i then $e\ \mathcal{R}\ req_i$

$rcwa_{req} = instanceOf(RCWA_j)$ and $RCWA_j \in S_{RCWA}$. $rcwa_{req}$ is an instance of the chosen $RCWA$, $RCWA_j$, which is used to encapsulate all certainty information and to determine whether the requirement req_i is sufficiently supported by the Requirement Elements contained in H_e.

Now that we have defined a Requirement and given a brief overview of Requirement Elements and $RCWAs$, we examine them in more detail. Section 5.4.1 focuses on Requirement Elements and section 5.4.2 focuses on $RCWAs$ and their associated instances.

5.4.1 Requirement Elements

Requirement Elements play an important role in Requirements and are represented by a tuple of the form $(CACQ_e, w_e, d_e)$. The tuple allows an investigator to capture the $CACQ$ which

supports a Requirement, the certainty weight (w_e) it contributes, should it hold, and a detailed description about the $CACQ$ and its associated certainty weight. The description element then allows an investigator to not only explain why $CACQ_e$ supports a particular Requirement, but to also explain why the $CACQ_e$ contributes the particular certainty weight. We define a Requirement Element, e_i, as follows:

$$\forall e_i \in S_e \text{ , } e_i = (CACQ_e, w_e, d_e) \text{ where}$$

$$CACQ_e \in S_{CACQ}$$

$w_e \in \mathbb{R}$ and w_e represents the certainty weight the requirement element contributes

$$d_e \in \mathbb{S}.$$

When an investigator adds Requirement Elements that support a particular Requirement, the weights they allocate to these Requirement Elements will however be based on the domain of weights specified in the $RCWA$ instance chosen for Requirement (D_{rcwa}). The Requirement Elements that support a particular Requirement, and that are hence an element of the Requirement's holder set (H_e) can therefore be defined as follows:

$$\forall req_i = (d_{req}, H_e, rcwa_{req}) \in S_{req} \text{ then}$$

$$\forall e_j \in H_e, e_j = (CACQ_e, w_e, d_e) \text{ where}$$

$$CACQ_e \in S_{CACQ}$$

$w_e \in D_{rcwa}$, such that $D_{rcwa} \subseteq \mathbb{R}$ and D_{rcwa} is the domain of weights which can be used with $rcwa_{req}$. w_e is the certainty weight that the Requirement Element e_j contributes to Requirement req_i

$d_e \in \mathbb{S}$, and d_e is a textual string explaining why $CACQ_e$ aids in supporting the requirement and why it contributes the allocated certainty weight, w_e.

Now that we have more formally defined a Requirement Element, we discuss a particular restriction that we have placed on Requirements and their associated holder set (H_e) that contains Requirement Elements. The restriction helps to ensure that H_e should contain a single Requirement Element for each distinct Container. We support this restriction by demonstrating that, should there exist two Requirement Elements which refer to the same Container in a Requirement, then they can be replaced by a single Requirement Element which incorporates both of their information. Our main reason for doing so is similar to the reasoning behind the restriction placed on the holder set of $CACQs$. A Requirement can be supported by a number of Requirement Elements, and the incorporation of such a restriction allows all information regarding a Container which supports the Requirement to be contained in a single Requirement Element. In this manner all information of how a Container supports a Requirement (namely its criteria), the certainty weight it contributes and the rationale behind it, is contained in a single location rather than distributed over several Requirement Elements. By doing so, an investigator's Finding Containers can be more clearly and

concisely captured, and their reasoning process can be captured more efficiently overall. An additional benefit is that the model can confirm whether Containers, their attributes and their criteria comply with the restrictions we previously specified in section 5.3.2 — particularly as all such information regarding a Container will be in a single $CACQ$. The new restriction can be more formally specified as follows:

$$\forall req_i = (d_{req}, H_e, rcwa_{req}) \in S_{req} \text{ then}$$

$$\forall e_j = (CACQ_{e_j}, w_{e_j}, d_{e_j}),\ e_k = (CACQ_{e_k}, w_{e_k}, d_{e_k}) \in H_e \text{ such that}$$

$$e_j \neq e_k \text{ and}$$

$$CACQ_{e_j} = (x_j, H_{ACQ_j}, d_{CACQ_j}) \text{ and } CACQ_{e_k} = (x_k, H_{ACQ_k}, d_{CACQ_k}) \text{ then}$$

$$x_j \neq x_k \text{ because if } x_j = x_k \text{ then}$$

$$\exists e_p = (CACQ_{e_p}, w_{e_p}, d_{e_p}) \in S_e \text{ such that}$$

$$CACQ_{e_p} = \left(x_p, H_{ACQ_p}, d_{CACQ_p}\right) \text{ where } x_p = x_j = x_k \text{ and } H_{ACQ_p} \text{ such that}$$

$$H_{ACQ_p} \text{ incorporates } H_{ACQ_j} \text{ and } H_{ACQ_k} \text{ as}$$

$$\forall\ ACQ_j = \left(a_j, \mathcal{C}_{s_j}, d_{ACQ_j}\right) \in H_{ACQ_j} \text{ and } ACQ_k = \left(a_k, \mathcal{C}_{s_k}, d_{ACQ_k}\right) \in H_{ACQ_k}$$

$$\text{where } a_j = a_k \text{ then}$$

$$\exists\ ACQ_p = \left(a_p, \mathcal{C}_{s_p}, d_{ACQ_p}\right) \in H_{ACQ_p} \text{ such that}$$

$$a_p = a_j = a_k \text{ and } \mathcal{C}_{s_p} = \mathcal{C}_{s_j} \cup \mathcal{C}_{s_k} \text{ and } d_{ACQ_p} \text{ incorporates the information from both } d_{ACQ_j} \text{ and } d_{ACQ_k} \text{ and}$$

$$\forall\ ACQ_j = \left(a_j, \mathcal{C}_{s_j}, d_{ACQ_j}\right) \in H_{ACQ_j} \text{ where}$$

$$\nexists\ ACQ_k = \left(a_k, \mathcal{C}_{s_k}, d_{ACQ_k}\right) \in H_{ACQ_k} \text{ such that } a_k = a_j \text{ then}$$

$$ACQ_j \in H_{ACQ_p}$$

and similarly

$$\forall\ ACQ_k = \left(a_k, \mathcal{C}_{s_k}, d_{ACQ_k}\right) \in H_{ACQ_k} \text{ where}$$

$$\nexists\ ACQ_j = \left(a_j, \mathcal{C}_{s_j}, d_{ACQ_j}\right) \in H_{ACQ_j} \text{ such that } a_j = a_k \text{ then}$$

$$ACQ_k \in H_{ACQ_p} \text{ and}$$

d_{CACQ_p} incorporates the information from both d_{CACQ_j} and d_{CACQ_k} hence

$CACQ_{e_j}$ and $CACQ_{e_k}$ can be replaced by $CACQ_{e_p}$ and

$w_{e_p} = w_{e_j} + w_{e_k}$ and d_{e_p} incorporates the information from both d_{e_j} and d_{e_k} hence

e_j and e_k can be replaced by e_p

In order to allow the model to evaluate the Requirement Element, we introduce another evaluation function, namely E_e, which takes a Requirement Element as a parameter and returns a Boolean value. The evaluation function performs its evaluation of the Requirement Element by evaluating the contained $CACQ$, with the use of E_{CACQ}. While E_e is simplistic, it plays an important role when evaluating a Requirement, and can be defined as follows :

$E_e\,(e_i) \rightarrow \mathbb{B}$, where

$e_i = (CACQ_e, w_e, d_e) \in S_e$ then E_e can be represented as

$E_e\,(e_i) = E_{CACQ}(CACQ_e)$.

5.4.2 Requirement Certainty Weighting Algorithms ($RCWAs$)

While we have provided a brief overview of $RCWAs$ and their importance in evaluating the Requirement Elements of a particular Requirement as well as the certainty they contribute towards a Requirement being met, our intention for this section is to go into more detail on $RCWAs$ and their instances. We discuss how $RCWAs$ and their instances are used, the benefits they offer and how they use Requirement Elements in order to provide dynamic feedback to an investigator with regards to their progress in achieving their desired certainty level for the Requirement.

We represent each $RCWA$ as a tuple of the form $(D_{RCWA}, T_{RCWA}, A_{RCWA}, d_{RCWA})$. D_{RCWA} is the particular domain of weights that can be assigned to Requirement Elements, should the particular $RCWA$ be chosen. T_{RCWA} is the domain of thresholds from which a threshold can be chosen to represent the investigator's desired certainty level. A_{RCWA} is the algorithm used to evaluate a Requirement and to hence determine whether the Requirement meets its certainty threshold. In order to evaluate a Requirement, A_{RCWA} takes three parameters: the set of all Requirement Elements that support the Requirement (contained in H_e of the Requirement), a particular threshold t (such that $t \in T_{RCWA}$), and a w_{total} in which A_{RCWA} stores the certainty total in order to reflect an investigator's certainty progress. The A_{RCWA} then uses the Requirement Elements contained in H_e along with their associated weights to determine whether the Requirement Elements meet the specified threshold. The total that the A_{RCWA} derives from the weights of the Requirement Elements is then stored in the variable w_{total}, allowing the model to provide feedback to the investigator on their progress in achieving the desired certainty threshold. The description element, d_{RCWA}, provides a means to capture the purpose of the particular $RCWA$ as well as how it performs its intended certainty check through the use of A_{RCWA}.

One example of a A_{RCWA}, which we refer to as $A_{RCWA_{Default}}$, iterates through the Requirement Elements and, should the Requirement Element hold and hence its $CACQ$

evaluate to true, then the algorithm adds the Requirement Elements certainty weight to a certainty total for the Requirement. Once $A_{RCWA_{Default}}$ has completed iterating through all the Requirement Elements it then sets w_{total} equal to the calculated certainty total and compares whether w_{total} meets the required certainty threshold, t, specified by the investigator. If w_{total} is greater than or equal to the threshold t, then the Requirement Elements sufficiently support their associated Requirement with the required level of certainty and the Boolean value $True$ is returned. Should the comparison not hold, then the Requirement Elements do not sufficiently support their associated Requirement with the desired level of certainty and the Boolean value $False$ is returned. Each A_{RCWA} therefore helps to encapsulate how the certainty check is performed, such as $A_{RCWA_{Default}}$ that is defined as follows :

$A_{RCWA_{Default}} =$

$boolean\ \ PerformEvaluation\ (H_e, t, w_{total}{}^{+})\ where$

$H_e \in S_e\ and\ t \in \mathbb{R}$ where $\mathbb{R}$ is the set of all real numbers and $w_{total}{}^{+}$ is modifiable

$\{\ Begin$

$sum = 0$

$Foreach\ (e_i = (CACQ_e, w_e, d_e)\ \in H_e)$

$\{Begin\ Foreach$

$IF\ (E_e(e_i) == true)$

$Then: sum\ =\ sum + w_e$

$End\ Foreach\}$

$set\ w_{total} =$ sum

$IF\ (w_{total} \geq\ t)$

$Then: H_e\ meets\ the\ specifed\ threshold\ and\ therefore\ return\ True$

$Else: H_e\ does\ not\ meet\ the\ specifed\ threshold\ and\ therefore\ return\ False$

$End\}$

Through the use of these algorithms, $RCWAs$ provide investigators with a means to dynamically evaluate their progress in an investigation, as well as the impacts various Requirement Elements have on their reasoning process. This can be shown in the example algorithm $A_{Default}$, which builds on the dynamic nature of the $CACQ$s and takes into account any modifications to the holder set of the Requirement and, hence, the associated Requirement Elements as well. Should the $CACQ$ of a Requirement Element no longer hold, and hence the associated Container no longer meet its required criteria (such as it is no longer admissible), then the certainty weighting algorithm will allow the investigator to see the impacts these changes have on the certainty of their Requirement, through (w_{total}). By

allowing the investigator to receive feedback about w_{total} and which Requirements do not meet the specified certainty threshold, the investigator can then take the necessary action to ensure that the required certainty level (t) is achieved. A further benefit of receiving feedback regarding w_{total} is that an investigator can dynamically see the impact on a Requirement when supporting Requirement Elements are added, allowing the model to provide progress feedback.

In order to ensure that the model is as flexible as possible, we chose to not limit A_{RCWA} to only $A_{Default}$. By doing so, algorithms that provide more diverse behaviour and provide greater flexibility can be created. These algorithms can first check the party responsible for a particular Container, and hence based on that party's credibility and some form of credibility criteria, determine whether the certainty weight should be added and/or only a portion of it. Another example is a particular algorithm that simply returns true, allowing an investigator to avoid having to use certainties. We refer to such a $RCWA$ as a Null $RCWA$. Now that we have discussed A_{RCWA}, we formally define a $RCWA$ and an instance of a $RCWA$.

In order to formally define a $RCWA$ we must first define the set of all $RCWAs$ as S_{RCWA}, and then we define each particular $RCWA$, denoted $RCWA_i$, as follows

$$\forall RCWA_i \in S_{RCWA}, RCWA_i = \left(D_{RCWA_i}, T_{RCWA_i}, A_{RCWA_i}, d_{RCWA_i}\right) \text{ such that}$$

$D_{RCWA_i} \subseteq \mathbb{R}$, where D_{RCWA_i} is the domain of weights which can be used in $RCWA_i$

$T_{RCWA_i} \subseteq \mathbb{R}$, and T_{RCWA_i} is the domain of thresholds which can be used in $RCWA_i$

$A_{RCWA_i} \in S_{A_{RCWA}}$, and A_{RCWA_i} is a particular certainty weighting algorithm such that

$$A_{RCWA_i}\left(H_e, t, {w_{total}}^{+}\right) \rightarrow \mathbb{B}, \text{ where}$$

$H_e \subseteq S_e$, $t \in T_{RCWA_i}$ and $w_{total} \in \mathbb{R}$ and w_{total} is modifiable and is used by A_{RCWA_i} in order to store the calculated certainty total that the Requirement Elements contained in H_e, contribute to a Requirement

$d_{RCWA_i} \in \mathbb{S}$ and d_{RCWA_i} describes the purpose and functioning of $RCWA_i$

An instance of a particular $RCWA$, which we denote as $rcwa$, can be seen as a specialisation of a $RCWA$ (as was previously discussed), inheriting the domain of the weights of the $RCWA$ and the certainty weighting algorithm A_{RCWA}. The instance also provides a storage facility for the certainty total, w_{total} and the certainty threshold chosen by the investigator. This allows the model to more easily provide feedback and to provide a storage facility for the manner in which certainty is calculated for the Requirement. In addition, the instance includes a description element that allows an investigator to capture why a particular $RCWA$ was chosen and why the particular certainty threshold was chosen for the Requirement. This particular instance (containing all certainty information regarding the Requirement) is then stored within the Requirement. An instance of a $RCWA$, denoted as $rcwa_j$, is defined as follows:

$$\forall RCWA_i = \left(D_{RCWA_i}, T_{RCWA_i}, A_{RCWA_i}, d_{RCWA_i}\right) \in S_{RCWA}$$

an instance of $RCWA_i$, denoted $rcwa_j$ can then be defined as

$$rcwa_j = \text{instanceOf}(RCWA_i)$$

then

$\forall rcwa_j = \text{instanceOf}(RCWA_i)$, $rcwa_j = (D_{rcwa_j}, t_{rcwa_j}, A_{rcwa_j}, d_{rcwa_j}, w_{total})$ such that

$$D_{rcwa_j} = D_{RCWA_i}$$

$$t_{rcwa_j} \in T_{RCWA_i}$$

$$A_{rcwa_j} = A_{RCWA_i}$$

$d_{rcwa_j} \in \mathbb{S}$, such that $d_{rcwa_j} \neq d_{RCWA_i}$ and d_{rcwa_j}describes why the instance of $RCWA_i$ was chosen and why the particular threshold, t_{rcwa_j}, was used

$w_{total} \in \mathbb{R}$, and w_{total} is used by A_{rcwa_j} to store the certainty total for a Requirement with which the instance $rcwa_j$ is directly associated.

5.4.2.1 Examples of Weighting Algorithms

In this section, we provide examples of possible $RCWA$s, describing how they can be used by an investigator to achieve their intended results. We first discuss the Percentage-Based $RCWA$ that allows an investigator to specify certainty weights for Requirement Elements and the required certainty level, t, as percentages. We then discuss the At Least N $RCWA$ that allows an investigator to specify the number of Requirement Elements that are required in order for a Requirement to be sufficiently supported. Finally, we discuss the Null $RCWA$ which allows an investigator to ignore certainties and simply capture their chain of reasoning.

5.4.2.1.1 Percentage-Based $RCWA$ ($RCWA_{PB}$)

The Percentage-Based $RCWA$ ($RCWA_{PB}$) is a possible $RCWA$ that uses percentages to convey the certainty weights that Requirement Elements contribute. The particular domain for such a $RCWA$ and the domain of thresholds is then $(0; 100]$. When an investigator uses an instance of the Percentage-Based $RCWA$ for a Requirement, the threshold, t, that they choose will reflect the particular percentage of certainty that the investigator requires for the Requirement to be sufficiently supported. $RCWA_{PB}$ can be more formally defined as follows

$$RCWA_{PB} \in S_{RCWA}$$

$$RCWA_{PB} = \left(D_{RCWA}, T_{RCWA}, A_{RCWA_{Default}}, d_{RCWA}\right) \text{ where}$$

$$D_{RCWA} = \{d \in \mathbb{R} \mid 0 < d \leq 100\}$$

$$T_{RCWA} = \{t \in \mathbb{R} \mid 0 < t \leq 100\}$$

d_{RCWA} = A Percentage-Based $RCWA$ allows an investigator to use percentages to capture the certainty that Requirement Elements contribute and the desired certainty level that must be attained for a Requirement to be sufficiently supported.

5.4.2.1.2 At Least N $RCWA$ ($RCWA_{ALN}$)

The At Least N $RCWA$ ($RCWA_{ALN}$) is another example of a potential $RCWA$ and it is used by an investigator to specify the number of Requirement Elements (N such that $N \in \mathbb{Z}^+$) that must hold for the Requirement to be sufficiently supported. This particular algorithm is beneficial when certain Requirements may play different roles (ranging in importance) in an investigator's reasoning process. The investigator can then specify the number of Requirement Elements that are needed for these Requirements to hold. $RCWA_{ALN}$ uses a domain $\{1\}$ for the weights of the Requirement Elements, and it uses the algorithm $A_{RCWA_{Default}}$ to iterate through the Requirement Elements in order to confirm that the specified certainty threshold is met. We can more formally define $RCWA_{ALN}$ as follows:

$$RCWA_{ALN} \in S_{RCWA} \text{ and}$$

$$RCWA_{ALN} = \left(D_{RCWA}, T_{RCWA}, A_{RCWA_{Default}}, d_{RCWA}\right) \text{ such that}$$

$$D_{RCWA} = \{1\} \text{ and}$$

$$T_{RCWA} = \mathbb{Z}^+$$

d_{RCWA}=At Least N $RCWA$ allows an investigator to specify the number of Requirement Elements that must hold in order for a Requirement to be sufficiently supported

5.4.2.1.3 Null $RCWA$ ($RCWA_{Null}$)

Earlier we discussed how, despite the benefits of $RCWAs$, they can be a hindrance for investigators, particularly if an investigator would simply like to capture their reasoning process without having to factor in their certainty. In order to cater for this an investigator could use a Null $RCWA$ ($RCWA_{Null}$), for which the domain of weights and the domain of thresholds is equal to the set $\{0\}$. The particular certainty weighting algorithm for Null $RCWA$ could most certainly be set to $A_{RCWA_{Default}}$, as the threshold will always be met (as $0 \geq 0$). However, we use this opportunity to discuss another possible algorithm that may be of use namely, $A_{RCWA_{Null}}$. $A_{RCWA_{Null}}$ is a very simplistic algorithm in that it simply returns true and avoids the cost of iterating through all the Requirement Elements.

$A_{RCWA_{Null}} =$

$boolean\ \ PerformEvaluation\ (H_e, t, w_{total})$

$\{ Begin$

$set\ w_{total} = 0;$

$return\ true;$

$End\}$

We can then formally define $RCWA_{Null}$ as follows

$$RCWA_{Null} \in S_{RCWA} \text{ and}$$

$$RCWA_{Null} = \left(D_{RCWA}, T_{RCWA}, A_{RCWA_{Null}}, d_{RCWA}\right) \text{ such that}$$

$$D_{RCWA} = \{0\}$$

$$T_{RCWA} = \{0\}$$

d_{RCWA}= Null $RCWA$ allows investigators to avoid having to factor in certainties into their reasoning process.

5.4.2.2 Summary of $RCWAs$

$RCWAs$ and their instances provide a means for investigators to factor certainty into their Requirements, should they wish to do so. They provide a means for investigators to specify the desired certainty level that Requirements must attain before they are sufficiently supported and to receive feedback on their progress in achieving the desired certainty level. These $RCWAs$ and their instances hence play an important role in the evaluation of a Requirement, and in section 5.4.3 we go into more detail on how this is performed.

5.4.3 The Evaluation of a Requirement

Requirements that are represented by the $(d_{req}, H_e, rcwa_{req})$ are composed of Requirement Elements of the form $(CACQ_e, w_e, d_e)$ but how are these Requirements evaluated? In order to evaluate a Requirement, req_i, we introduce the evaluation function E_{Req}, which uses the $RCWA$ instance $(rcwa_{req})$ of req_i in order to invoke its certainty weighting algorithm. E_{Req} passes the holder set of Requirement Elements from req_i, the threshold specified in $rcwa_{req}$ and the w_{total} variable contained in $rcwa_{req}$, as parameters to the certainty weighting algorithm and returns its Boolean result. The Boolean result represents whether or not the Requirement is supported with the specified level of certainty according to the chosen $RCWA$ instance and is represented as follows :

$$E_{Req}(req_i) \rightarrow \mathbb{B} \text{ where}$$

$$req_i = \left(d_{req}, H_e, rcwa_{req}\right) \in S_{req} \text{ and}$$

$$rcwa_{req} = (D_{rcwa_j}, t_{rcwa_j}, A_{rcwa_j}, d_{rcwa_j}, w_{total}) \text{ then}$$

E_{Req} can be represented as

$$E_{Req}\,(req_i) = A_{rcwa_j}(\,H_e, t_{rcwa_j}, {w_{total}}^{+}) \rightarrow \mathbb{B}\,.$$

5.4.4 A Summary of a Requirement

A Requirement allows an investigator to represent a condition that must be met in order for the investigator to provide a confirmation for a particular Argument's conclusion. Investigators are then able to capture various $CACQs$ that support the Requirement, the

certainty weight, the rationale behind why the $CACQ$ supports the Requirement and why the $CACQ$ shows the Requirement holds with a particular certainty, through the use of Requirement Elements. Requirements also include an instance of a $RCWA$ that allows investigators to capture the particular domain of weights that were used to allocate certainty contributions to the $CACQs$. In addition, the instance allows an investigator to capture the particular certainty threshold that the investigator believes the Requirement must attain before they are satisfied that the Requirement is sufficiently supported. The instance of a $RCWA$ also allows an investigator to capture and encode the manner in which the certainty contributions of the $CACQs$ (provided they hold) is used to determine whether the Requirement's threshold is met. Incorporating such information into the Requirement allows us to capture an investigator's reasoning process in greater detail, allowing such information to be easily accessible when it is needed and helping to aid investigators who want to continue an investigation. We now explore how these Requirements are used to formulate confirmations in an investigator's Supporting and Refuting Arguments.

5.5 An Argument — Supporting and Refuting

We discussed Requirements and how they are composed of $RCWAs$ and various Requirement Elements that support the Requirement. We also discussed how each of the Requirement Elements consists of a $CACQ$ that allows an investigator to capture the various criteria that a Container must comply with to serve its role in an investigation. We now look at how these Requirements and their smaller components are used in order to help an investigator form two types of Arguments in the Finding Container, namely the Supporting Argument and the Refuting Argument.

In this dissertation, we have emphasized how an investigator must capture their hypotheses and take into account not only inculpatory evidence but also exculpatory evidence and evidence of tampering in order to form sound conclusions. We also discussed how an investigator should take into account alternate explanations, explaining why they cannot hold. The Finding Container we propose therefore contains a hypothesis, a Supporting Argument and a Refuting Argument (as discussed in section 5.2), among other elements. The hypothesis is the statement the investigator wants to support or refute, and the Supporting Argument and the Refuting Argument encapsulate the means to do so, along with a particular conclusion with regards to the hypothesis, should the Argument hold. The Supporting Argument contains a conclusion that supports the hypothesis, and the Refuting Argument contains a conclusion that refutes the hypothesis. The investigator then uses Requirements to capture when these Arguments should hold and to *argue* when their associated conclusion should be derived.

Requirements play an important role in these Arguments because they serve as conditions that an investigator believes must be met in order for the Argument's conclusion to be derived. In our model, we allow investigators to capture the combination of conditions that must be met to provide a confirmation for the Argument's conclusion. We allow the investigator to combine these Requirements using logical connectives and brackets, so that the combination is captured more formally and with less ambiguity. We also allow the

investigator to capture multiple confirmations, to circumvent legal outcomes which are inconsistent with logical deductions, within these Arguments. A Supporting Argument therefore allows an investigator to capture the various combinations of Requirements that, should they be met, provide confirmations for why the hypothesis should be supported and why the supporting conclusion should hold. In contrast the Refuting Argument contains the various combinations of Requirements that, should they be met, provide confirmations for why the hypothesis should be refuted and why the refuting conclusion should hold.

These combinations of Requirements (provided they hold) can allow the investigator to confirm the Argument's conclusion with a particular certainty, based on the form of proof they provide. In order to allow an investigator to factor in the certainty that these confirmations provide for the Argument's conclusion, an approach similar to that incorporated into Requirements is used. The use of a similar approach provides consistency for the model with regards to how certainty is incorporated into its Finding Components. We incorporate and allow an investigator to capture such behaviour through the use of two new components, namely Combination Certainty Weighting Algorithms ($CCWAs$) and their instances. These $CCWAs$ and their instances resemble $RCWAs$ and their instances in purpose and in structure, except they apply to Arguments and combinations of Requirements which serve as confirmations, rather than Requirements and Requirement Elements.

Each of the combinations of Requirements (which provide a confirmation should they be met) is associated a certainty weight based on the certainty with which it can show that the Argument's conclusion holds. The investigator can then capture a certainty threshold that the Argument must attain before the investigator is satisfied that the Argument's conclusion holds. All certainty information regarding the Argument that an investigator specifies, the domain of weights that can be associated to these confirmations, the certainty threshold chosen for the Argument, the calculated certainty total and the manner in which the certainty total is calculated and compared against the certainty threshold, is contained within a $CCWA$ instance (similar to a $RCWA$ instance). The $CCWA$ instance is then stored within the Argument itself, allowing such information to be encapsulated within the Argument in a well-defined structure.

Capturing a detailed reasoning process is an important goal for our dissertation and therefore we also want to capture not only the combination of Requirements that provide confirmation as well as their certainty weights, but also the rationale behind them. Investigators should be able to capture why they believe the combination of Requirements provides a confirmation for an Argument's conclusion, and why the investigator believes it does so with the allocated certainty. In order to allow investigators to capture and encapsulate all such information regarding the combination of Requirements, we introduce a new Finding Component, namely the Requirement Combination ($ReqC$). These Requirement Combinations are then captured within the Argument, along with the $CCWA$ instance and the particular conclusion.

We represent an Argument (denoted Arg), Supporting and Refuting, and capture all the aspects we discussed, through a tuple of the form $(H_{ReqC}, ccwa, d_{conclusion})$. H_{ReqC} is a holder set for all Requirement Combinations ($ReqCs$) that, should they be met, each provide

confirmation for the Argument's conclusion, $d_{conclusion}$. Capturing all this information within the Argument allows the investigator to capture why they believe the conclusion can/cannot be derived, making such information available when it is needed. The second element of the Argument tuple, namely $ccwa$, is an instance of the $CCWA$ that captures all certainty information regarding the Argument. Before we can more formally define an Argument, we must first define the set of all Arguments as S_{Arg}. We can then define each particular Argument, Arg_i, as an element of the set such that

$$Arg_i \in S_{Arg}$$

We then define each particular Argument, Arg_i, as follows:

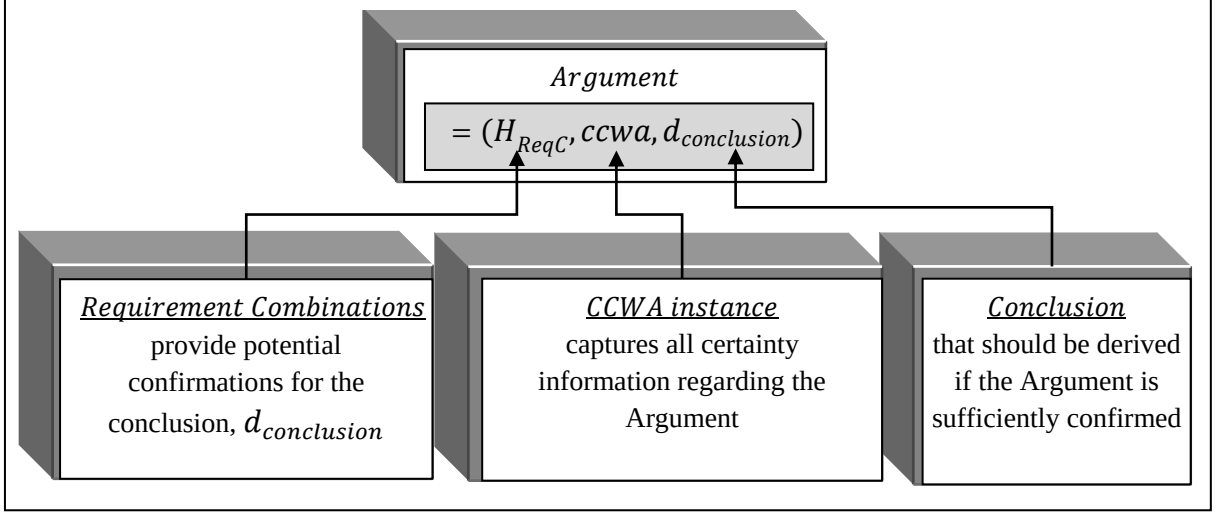
$$\forall Arg_i \in S_{Arg},\ Arg_i = (H_{ReqC}, ccwa_{Arg}, d_{conclusion}) \text{ such that}$$

$H_{ReqC} \subseteq S_{ReqC}$ where S_{ReqC} is defined as the set of all Requirement Combinations and H_{ReqC} is a holder set for all Requirement Combinations which provide confirmation for the particular conclusion $d_{conclusion}$

$$ccwa_{Arg} = instanceOf(CCWA_{Arg}) \text{ where } CCWA_{req} \in S_{CCWA}$$

$d_{conclusion} \in \mathbb{S}$ and $d_{conclusion}$ describes the particular conclusion that an investigator believes should be made should the Argument, Arg_i hold.

Figure 9: An Argument



We explore these Requirement Combinations and the holder set in more detail in section 5.5.1 and then the $CCWA$ instance in section 5.5.2. Once we have completed our detailed discussion on the components, we discuss how the Argument is evaluated dynamically through the use of the certainty weighting algorithm contained in the $CCWA$ instance.

5.5.1 A Requirement Combination ($ReqC$) and the Holder Set (H_{ReqC})

A Requirement Combination is represented in the model by a tuple of the form $(combination_{req}, w_{ReqC}, d_{ReqC})$. The first element of the tuple, $combination_{req}$, is a

combination of Requirements that are combined using logical connectives and brackets which, should they hold, provide a confirmation for an Argument's conclusion confirming why the particular conclusion can be derived. The w_{ReqC} is used to capture the particular certainty that the Requirement Combination contributes to confirming the particular Argument's conclusion. The description element, d_{ReqC}, is used to capture the rationale behind the combination of Requirements, why the combination provides confirmation for the Argument's conclusion and why it contributes the particular certainty weight. In order to ensure that $combination_{req}$ is defined formally and without ambiguity, the approach we chose was to define such a combination through the use of a grammar.

A grammar is a set of rules that describes how sentences of a language are formed [60][65]. We use a grammar to formally specify the set of rules of how Requirements can be combined when forming a Requirement Combination, so as to avoid ambiguity with how Requirements can be combined. The grammar (G) consists of non-terminals (N), terminals (T), a starting symbol (S) and a set of production rules (P) and can be represented as follows:

$$G = (N, T, S, P) \text{ where}$$

$$N = \{RequirementCombination,\ OrRequirement, AndRequirement,\ NotRequirement,\ BracketRequirement,\ Requirement\}$$

$$T = S_{req} \cup \{\vee, \wedge, \sim, (,)\}$$

$$S = RequirementCombination$$

$$\begin{aligned} P = \{ \quad & RequirementCombination && = && OrRequirement. \\ & OrRequirement && = && OrRequirement \vee AndRequirement \\ & && && \mid AndRequirement. \\ & AndRequirement && = && AndRequirement \wedge NotRequirement \\ & && && \mid NotRequirement. \\ & NotRequirement && = && \sim BracketRequirement \\ & && && \mid BracketRequirement. \\ & BracketRequirement && = && (RequirementCombination) \\ & && && \mid Requirement. \\ & Requirement && = && req_i, \text{ where } req_i \in S_{req}. \\ \} & \end{aligned}$$

We then define the set of all $combination_{req}$, which can be produced by Requirements combined with logical connectives and brackets, as $S_{combination_{req}}$. We then define each particular $combination_{req}$, namely $combination_{req_i}$, as an element of this set such that

$$combination_{req_i} \in S_{combination_{req}}.$$

Now that we have discussed and defined the combination of Requirements that an investigator can use to support why an Argument's conclusion should be derived, we define the Requirement Combination. Each possible Requirement Combination ($ReqC_i$), which is an element of the set of all Requirement Combinations (denoted S_{ReqC}) can be defined as follows:

$$\forall ReqC_i \in S_{ReqC},\ ReqC_i = (combination_{req}, w_{ReqC}, d_{ReqC}) \text{ where}$$

$$combination_{req} \in S_{combination_{req}}$$

$$w_{ReqC} \in \mathbb{R}$$

$$d_{ReqC} \in \mathbb{S}.$$

The Requirement Combinations that provide confirmations for a particular Argument's ($Arg_i's$) conclusion are then collectively stored in the holder set H_{ReqC} of the Argument, allowing all confirmations for a Argument to be captured in a single location. These Requirement Combinations are then allocated weights that are based on the $CCWA$ instance chosen for the Arg_i, or more precisely the domain of certainty weights associated with the $CCWA$ instance. The domain of weights of a $CCWA$ instance, in a similar manner to a $RCWA$ instance, represents the possible values that can be allocated to Requirement Combinations when using the particular $CCWA$. We can therefore define the Requirement Combination for an Argument, Arg_i as follows:

$$\forall Arg_i = (H_{ReqC}, ccwa_{Arg}, d_{conclusion}) \in S_{Arg} \text{ then}$$

$$\forall ReqC_j \in H_{ReqC}, ReqC_j = (combination_{req}, w_{ReqC}, d_{ReqC}) \text{ then}$$

$combination_{req} \in S_{combination_{req}}$ and $combination_{req}$ represents the combination of Requirements combined using logical connectives and brackets which provide confirmations for conclusion, $d_{conclusion}$

$w_{ReqC} \in D_{ccwa_{Arg}}$, where w_{ReqC} is the particular certainty weight that the Requirement Combination , $ReqC_j$, contributes to Argument Arg_i and $D_{ccwa_{Arg}}$ is the domain of all possible weights that can be used in $ccwa_{Arg}$

$d_{ReqC} \in \mathbb{S}$ and d_{ReqC} is a textual string explaining why $combination_{req}$ serves as a confirmation for $d_{conclusion}$ and why it was allocated the particular certainty weight, w_{ReqC} .

In order to evaluate a Requirement Combination we introduce an evaluation function, E_{ReqC}, which takes a single parameter, namely a Requirement Combination, and produces a Boolean value. E_{ReqC} can therefore be represented as follows :

$$E_{ReqC}(ReqC_i) \rightarrow \mathbb{B} \text{ where}$$

$$ReqC_i \in S_{ReqC}.$$

E_{ReqC} evaluates a Requirement Combination by iterating through its Requirements and evaluating them using E_{req}. E_{ReqC} then combines the evaluation results for the Requirements, according to the brackets and operators used in the combination and their order of precedence, in order to produce its output, namely a Boolean value. The $CCWA$ instance associated with an Argument then relies on these evaluations of the Requirement Combinations in order to determine whether the conclusion is sufficiently supported.

5.5.2 Combination Certainty Weighting Algorithm ($CCWA$)

The Combination Certainty Weighting Algorithm ($CCWA$) is similar to a $RCWA$ in terms of structure and functionality, however it applies to Requirement Combinations and Arguments instead of Requirement Elements and Requirements. A $CCWA$ is represented in this dissertation by a tuple of the form $(D_{CCWA}, T_{CCWA}, A_{CCWA}, d_{CCWA})$ that has a domain of weights (D_{CCWA}), a domain of thresholds (T_{CCWA}), a particular certainty weighting algorithm (A_{CCWA}) and finally a description (d_{CCWA}). The domain of thresholds is used by an investigator to select a particular threshold, t, such that $\in T_{CCWA}$, resembling the certainty level an Argument must attain before the investigator is satisfied that its conclusion is sufficiently supported. The algorithm A_{CCWA} then uses the Requirement Combinations and their associated weights to determine whether the certainty level, t, is met and hence whether the Argument's conclusion is sufficiently supported. The description element d_{CCWA} is used to capture information about the $CCWA$, its purpose and how its associated A_{CCWA} functions. The direct resemblance between $CCWAs$ and $RCWAs$ means we will not go into great detail about their components, and is one of the benefits with using consistency in the model.

In order to formally define a $CCWA$, we must first define the set of all $CCWAs$ as S_{CCWA}. We then define each particular $CCWA$, denoted as $CCWA_i$, as follows:

$$\forall CCWA_i \in S_{CCWA}, CCWA_i = \left(D_{CCWA_i}, T_{CCWA_i}, A_{CCWA_i}, d_{CCWA_i}\right) \text{ such that}$$

$D_{CCWA_i} \subseteq \mathbb{R}$, where D_{CCWA_i} is the domain of weights which can be used in $CCWA_i$

$T_{CCWA_i} \subseteq \mathbb{R}$, and T_{CCWA_i} is the domain of thresholds which can be used in $CCWA_i$

$A_{CCWA_i} \in S_{A_{CCWA}}$, and A_{CCWA_i} is a particular certainty weighting algorithm such that

$$A_{CCWA_i}\left(H_{ReqC}, t, {w_{total}}^{+}\right) \rightarrow \mathbb{B}, \text{ where}$$

$H_{ReqC} \subseteq S_{ReqC}$, $t \in T_{CCWA_i}$ and $w_{total} \in \mathbb{R}$ and w_{total} is modifiable and is used by A_{CCWA_i} in order to store the calculated certainty total that the Requirement Combinations contained in H_{ReqC} contribute to an Argument

$d_{CCWA_i} \in \mathbb{S}$ and d_{CCWA_i} describes the purpose and functioning of $CCWA_i$

An instance of a $CCWA$ is also similar to that of a $RCWA$ and can be seen as a specialisation of a $CCWA$ that focuses on storing the required certainty level for the argument, t, and the calculated certainty total (w_{total}) for the Argument with which the instance is associated. The

instance of the $CCWA$ inherits the domain of certainty weights and the certainty weighting algorithm, which is used to confirm that the Argument meets its required certainty level through the use of the Requirement Combinations contained in H_{ReqC}. The description element of a $CCWA$ instance is used to capture why an investigator chose a particular $CCWA$ and why the particular certainty threshold, t, was chosen. An instance of a $CCWA$, denoted as $ccwa_j$, can be defined as follows:

$\forall CCWA_i = (D_{CCWA_i}, T_{CCWA_i}, A_{CCWA_i}, d_{CCWA_i}) \in S_{CCWA}$, an instance of $CCWA_i$, denoted $ccwa_j$, can be represented as follows:

$$ccwa_j = \text{instanceOf}(CCWA_i)$$

then

$\forall ccwa_j = \text{instanceOf}(CCWA_i)$, $ccwa_j = (D_{ccwa_j}, t_{ccwa_j}, A_{ccwa_j}, d_{ccwa_j}, w_{total})$ such that

$$D_{ccwa_j} = D_{CCWA_i}$$

$$t_{ccwa_j} \in T_{CCWA_i}$$

$$A_{ccwa_j} = A_{CCWA_i}$$

$d_{ccwa_j} \in \mathbb{S}$, such that $d_{ccwa_j} \neq d_{CCWA_i}$ and d_{ccwa_j}describes why the instance of $CCWA_i$ was chosen and why the particular threshold, t_{ccwa_j}, was used

$w_{total} \in \mathbb{R}$, and w_{total} is used by A_{ccwa_j} to store the certainty total for an Argument with which the instance $ccwa_j$ is directly associated.

5.5.3 The Evaluation of an Argument

The evaluation of an Argument strongly relies on the $ccwa$ instance contained within the Argument and its inherited certainty weighting algorithm for Requirement Combinations. In order to perform the evaluation of an Argument, Arg_i, an evaluation function, E_{Arg}, is used. It performs the evaluation by using the $CCWA$ instance ($ccwa_{Arg}$) contained in Arg_i to invoke the certainty weighting algorithm for Requirement Combinations. E_{Arg} then passes the holder set of Requirement Combinations contained in Arg_i, the threshold value (t) specified in $ccwa_{Arg}$ and the certainty total variable contained in $ccwa_{Arg}$ as parameters, returning a Boolean value. The Boolean value that is returned represents whether the Argument's conclusion is sufficiently supported in terms of the required certainty specified in t. Overall, the evaluation of an Argument is similar to the evaluation of a Requirement using E_{Req}, and can be defined as follows :

$$E_{Arg}(Arg_i) \rightarrow \mathbb{B} \text{ where}$$

$$Arg_i = (H_{ReqC}, ccwa_{Arg}, d_{conclusion}) \in S_{Arg} \text{ and}$$

$ccwa_{Arg} = (D_{ccwa_j}, t_{ccwa_j}, A_{ccwa_j}, d_{ccwa_j}, w_{total})$ then

E_{Arg} can be represented as

$E_{Arg}\ (Arg_i) = A_{ccwa_j}(\ H_{ReqC}, t_{ccwa}, {w_{total}}^{+}) = b$ where $b \in \mathbb{B}$.

The evaluation of an Argument allows an investigator to easily see whether their Argument holds and hence whether its associated conclusion can be derived. Let us now examine how these Arguments, Supporting and Refuting, are factored into the Finding Container.

5.6 A Finding Container

A Finding Container (briefly discussed at the beginning of section 5.2) is what we have been building up to, namely providing a means for an investigator to capture their hypotheses, their conclusions, how they came to the conclusions and the Arguments, Supporting and Refuting, behind these conclusions. The Finding Container resembles the Evidence Container in order to reap its benefits and to more easily communicate the Finding Container. The Finding Container therefore contains a tag file, a Proof of Action History and the Finding Container Unit. The tag file for a Finding Container incorporates identification information (among other information), allowing investigators to refer to the Container and to allow the Container to be referenced (and embedded) in other Containers. We wanted the Finding Container to be able to capture an investigator's steps so that they cannot easily refute their involvement in a step, and to allow for a more detailed reconstruction of their reasoning process. The Proof of Action History (Chapter 4) and their contained Proof of Actions (Chapter 3) provide such a means and they also maintain a timeline of when these events took place. By maintaining a Proof of Action History for the Finding Container, a detailed audit trail for how the Finding Container came to be can be maintained, while reducing the administrative burden of the investigator having to record all such information.

The Finding Container Unit, similar to the Container Unit in an Evidence Container, includes attributes and the actual Container content, namely the investigator's finding and reasoning behind it. These attributes are used for capturing various characteristics of the finding and we previously discussed two such attributes in section 5.2. These attributes were the supported and the refuted attribute, which are Boolean values, allowing one to easily see whether a finding is supported, refuted or neither supported nor refuted. Before we discuss the Finding Container Unit in more detail we must first discuss a new Finding Component that plays an important role in the Finding Container Unit. We refer to this Finding Component as the Reasoning Algorithm.

The Reasoning Algorithm allows investigators to capture and encode how they reasoned using the Supporting and Refuting Arguments and to capture which Argument should be favoured, should both Arguments meet their certainty thresholds. We incorporate such a Finding Component as it provides a non-volatile storage for such information, helping others to understand how such conflicts were handled and reasoned about by the investigator. The Reasoning Algorithm also provides the model with the capability to use the investigator's

encoded reasoning process to dynamically evaluate their findings and determine whether a Finding Container's hypothesis should be supported or refuted. Should more contributions be made to the Finding Container, its Supporting and Refuting Arguments, or should changes result in certain Requirements no longer being able to serve their role in the confirmations, the model can then automate an investigator's reasoning. The automation of an investigator's reasoning helps to allow the Finding Container and the model in general to provide feedback more efficiently to an investigator on the impacts of their contributions and changes in an investigation.

A Reasoning Algorithm (RA) encapsulates an algorithm (A_{RA}) and a description (d_{RA}). The description is used to describe how A_{RA} works and what its purpose is. The Algorithm takes the Finding Container's Supporting Argument, its Refuting Argument, its supported attribute, its refuted attribute and its conclusion as parameters. The Reasoning Algorithm is then used to evaluate the Supporting Argument and the Refuting Argument and use the encoded reasoning to determine whether the Finding Container's hypothesis is sufficiently supported, refuted or neither sufficiently supported nor refuted. Depending on the Algorithm's reasoning, it then sets the supported attributed, the refuted attribute and the conclusion of the Finding Container, which are all passed as parameters. The Reasoning Algorithm performs its reasoning by first evaluating the Supporting Argument and the Refuting Argument, from which there may be four possible scenarios:

- In the first scenario, only the Supporting Argument is sufficiently supported and meets the required certainty threshold. The Reasoning Algorithm then sets the Finding Container's conclusion to the conclusion of the Supporting Argument and it sets the Finding Container's supported attribute to true and the refuted attribute to false.
- The second scenario is similar to the first, however, only the Refuting Argument is sufficiently supported and meets the required certainty threshold. The Reasoning Algorithm then sets the Finding Container's conclusion to the conclusion of the Refuting Argument and it sets the Finding Container's refuted attribute to true and the supported attribute to false.
- In the third scenario, neither the Supporting Argument nor the Refuting Argument are sufficiently supported and hence neither of the Arguments meet their required certainty thresholds. The Reasoning Algorithm then sets the Finding Container's conclusion to the empty string, showing that no conclusion can be made at this particular point in time with the required certainty. The Reasoning Algorithm also sets the supported attribute and the refuted attribute of the Finding Container to false, showing that the Finding Container's hypothesis cannot be supported or refuted at this point in time.
- In the fourth scenario, both the Supporting Argument and the Refuting Argument are sufficiently supported and hence meet their required certainty thresholds. This particular scenario requires a more in depth discussion.

The final scenario, where both the Supporting Argument and the Refuting Argument are supported, plays an important role in the purpose of Reasoning Algorithms, and is based on the concept of conflict resolution used in expert systems (Chapter 2). Each particular

Reasoning Algorithm can handle such a situation differently and, depending on how they handle this particular situation, will determine when an investigator chooses to use them for a Finding Container. Certain Reasoning Algorithms can for example decide to simply favour the Supporting Argument over the Refuting Argument in such a scenario, or vice versa. Should the Supporting and Refuting Argument use the same $CCWA$ for their instances, then the decision can be based on which Argument has the greatest certainty total, or it can be based on which Argument has the greatest certainty threshold. By allowing each Finding Container to be configured with a Reasoning Algorithm that suits the investigator's needs, investigators can choose the approach that provides the best dynamic feedback for each of their Finding Containers. Should an investigator not find a Reasoning Algorithm that suits their needs they can add a new Reasoning Algorithm to the model, which is then stored within a set, S_{RA}, making accessing and adding of all Reasoning Algorithms easier. In order to define a Reasoning Algorithm, RA_i, we first define the set of all Reasoning Algorithms as S_{RA}. We then define RA_i as follows:

$$\forall RA_i = (A_{RA_i}, d_{RA_i}) \in S_{RA} \text{ then}$$

$A_{RA_i} \in S_{A_{RA}}$ where $S_{A_{RA}}$ is the set of all algorithms that can be used within a Reasoning Algorithm and

$d_{RA_i} \in \mathbb{S}$, and d_{RA_i} provides a description of how A_{RA_i} , works, allowing investigators to find a Reasoning Algorithm that best suits their needs.

Investigators should also capture why they chose a particular Reasoning Algorithm. In order to do so and to ensure consistency with how algorithms are integrated into a Finding Container, we introduce an instance of a Reasoning Algorithm, denoted as ra. An instance of a Reasoning Algorithm is a specialisation of a Reasoning Algorithm inheriting the Algorithm A_{RA} but it incorporates a description element, which allows an investigator to describe why they chose a particular Reasoning Algorithm. The contained A_{ra} is then executed whenever a change is made to a Finding Container, whenever one of its building blocks is altered and/or no longer meets its specified criteria, and whenever execution is requested. A Reasoning Algorithm instance can be defined as follows:

$\forall RA_i = (A_{RA_i}, d_{RA_i}) \in S_{RA}$ then an instance of RA_i, denoted ra_j, can be represented as:

$$ra_j = \text{instanceOf}(RA_i)$$

Then

$$\forall ra_j = \text{instanceOf}(RA_i), ra_j = (A_{ra_j}, d_{ra_j}) \text{ such that}$$

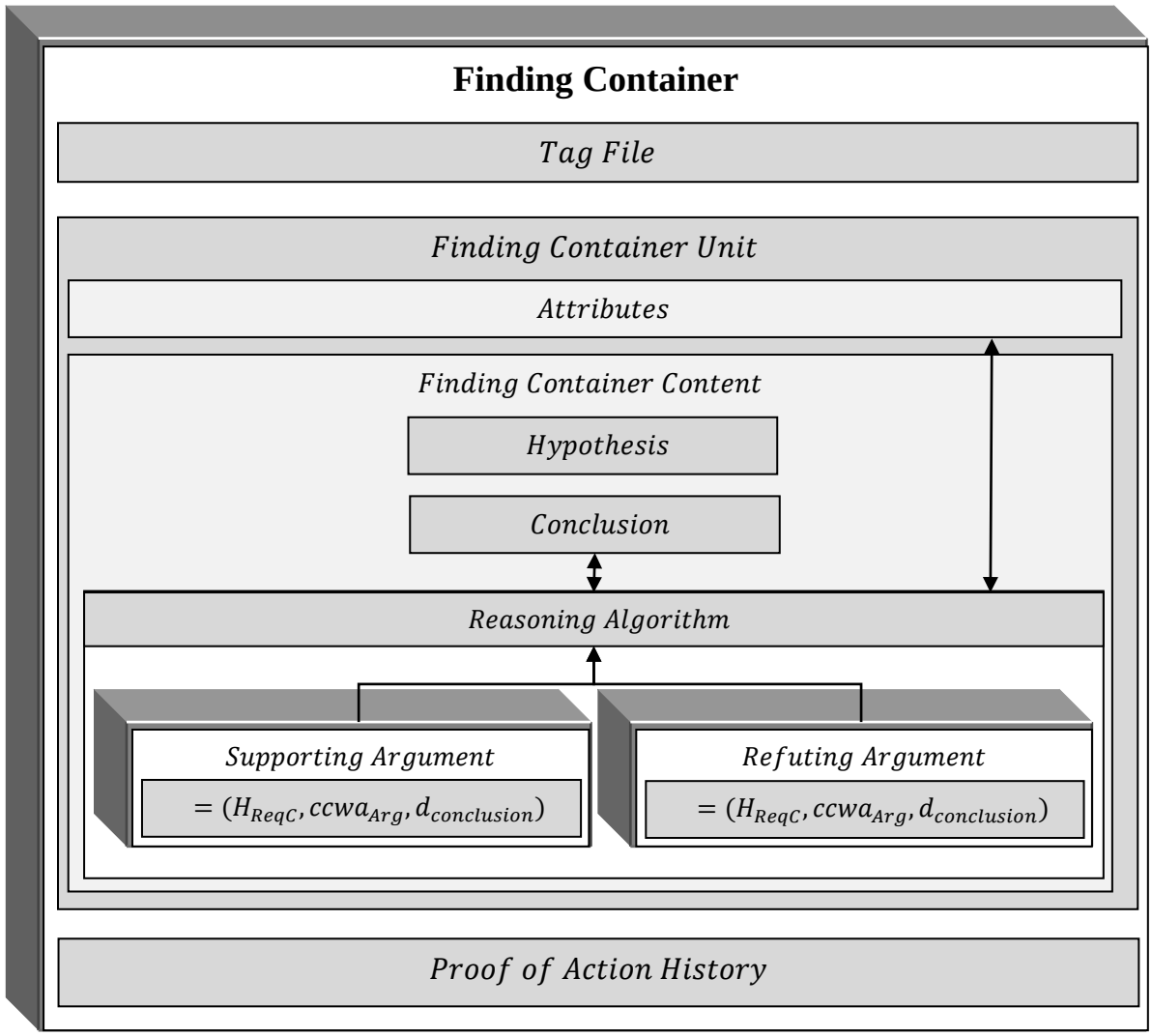
$$A_{ra_j} = A_{RA_i}$$

$d_{ra_j} \in \mathbb{S}$, such that, $d_{ra_j} \neq d_{RA_i}$, and d_{ra_j}describes why the investigator chose the particular Reasoning Algorithm for their Finding Container.

We now explore the Finding Container Unit more closely.

5.6.1 A Finding Container Unit

Figure 10: A Finding Container



The structure chosen for the Finding Container is demonstrated in Figure 10 and of particular interest for this section is the Finding Container Unit. The Finding Container Unit consists of set of attributes ($\mathbb{A}_x$) and the Finding Container Content. The Finding Container Content consists of a hypothesis or goal for the Finding Container ($d_{hypothesis}$), a Supporting Argument ($Arg_{supporting}$), a Refuting Argument ($Arg_{refuting}$), a Reasoning Algorithm instance (ra), and a conclusion ($d_{conclusion}$). The hypothesis is a particular statement that an investigator would like to support or refute during an investigation, and it is used to capture an investigator's main goal and focus for the particular Finding Container. The Supporting Argument is the particular Argument that allows an investigator to justify why the hypothesis holds and should be supported, through the many confirmations it contains. The Refuting Argument allows an investigator to capture why a hypothesis should not hold, allowing an investigator to capture their objectivity in the investigation. The set of attributes, contains the various attributes that are associated with the Finding Container, including two particular

attributes, namely the supported attribute and the refuted attribute. The Reasoning Algorithm uses the Supporting and Refuting Argument, and all of the investigator's encoded reasoning information, in order to *reason* about the conclusion it should derive and to set the values of the supported and refuted attribute accordingly. This is shown by the arrows from the Supporting and Refuting Arguments in Figure 10, and the attributes to the Reasoning Algorithm.

We define a Finding Container, denoted x_f, as an element of the set of all Finding Containers ($\mathbb{C}_F$) such that

$\forall x_f \in \mathbb{C}_F, x_f$ contains a Finding Container Unit, FCU, where

$$FCU = \left(\mathbb{A}_{x_f}, FCC \right) \text{ such that}$$

$\mathbb{A}_{x_f} \subseteq \mathbb{A}$ and $\mathbb{A}_{x_f}$ is the set of all attributes contained in x_f

and FCC is the Finding Container Content such that

$$FCC = (d_{hypothesis}, Arg_{supporting}, Arg_{refuting}, ra, d_{conclusion}) \text{ where}$$

$d_{hypothesis} \in \mathbb{S}$ and $d_{hypothesis}$ is a textual string stating what the hypothesis for Finding Container x_f is

$Arg_{supporting} \in S_{Arg}$ and $Arg_{supporting}$ is the Supporting Argument encapsulating all information for why the hypothesis, $d_{hypothesis}$, should be supported

$Arg_{refuting} \in S_{Arg}$ and $Arg_{refuting}$ is the Refuting Argument encapsulating all information for why the hypothesis, $d_{hypothesis}$, should be refuted

ra = instanceOf(RA_i) and where $RA_i \in S_{RA}$ is the set of all RAs and RA is the Reasoning Algorithm that *reasons* whether the hypothesis, $d_{hypothesis}$, is supported, refuted or neither supported nor refuted, based on $Arg_{supporting}$ and $Arg_{refuting}$

$d_{conclusion} \in \mathbb{S}$ and $d_{conclusion}$ is a textual string capturing what conclusion can be derived for the Finding Container at the current point in time.

5.6.2 A Summary of the Finding Container

Finding Containers allow investigators to capture their hypotheses and to capture their reasoning behind why the hypothesis is supported (Supporting Argument) and why the hypothesis is refuted (Refuting Argument). These Arguments allow investigators to demonstrate that they reasoned objectively and did not only factor in inculpatory evidence. Finding Containers also provide a means for investigators to encode how they reasoned about the hypothesis (Reasoning Algorithm), using these Arguments, allowing investigators to capture how they came to their conclusions. Each of these Arguments is composed of various potential confirmations consisting of combinations of Requirements. These combinations of Requirements allow investigators to capture what conditions must be met for them to provide

a confirmation. In this manner, an investigator can capture more in-depth information about the confirmation and the premises behind it. Investigators can also factor in the certainty that these confirmations provide through the use of $CCWAs$.

The investigator then uses various $CACQs$ to support why the particular Requirement is met. Each $CACQ$ allows an investigator to capture not only the Container that supports a Requirement, but also how (criteria) and why it does so (description element). Furthermore, investigators may factor in the certainty that these $CACQs$ contribute towards a Requirement being met through the use of $RCWAs$. The Finding Container therefore provides a standardised, yet flexible means for allowing an investigator to capture their reasoning — the manner in which it is formed and the rationale behind it — for each particular conclusion they derive. The Finding Container and its various Finding Components then encapsulate this information, allowing interested parties to view an investigator's reasoning at the granularity that suits their needs.

5.7 The Chain of Findings

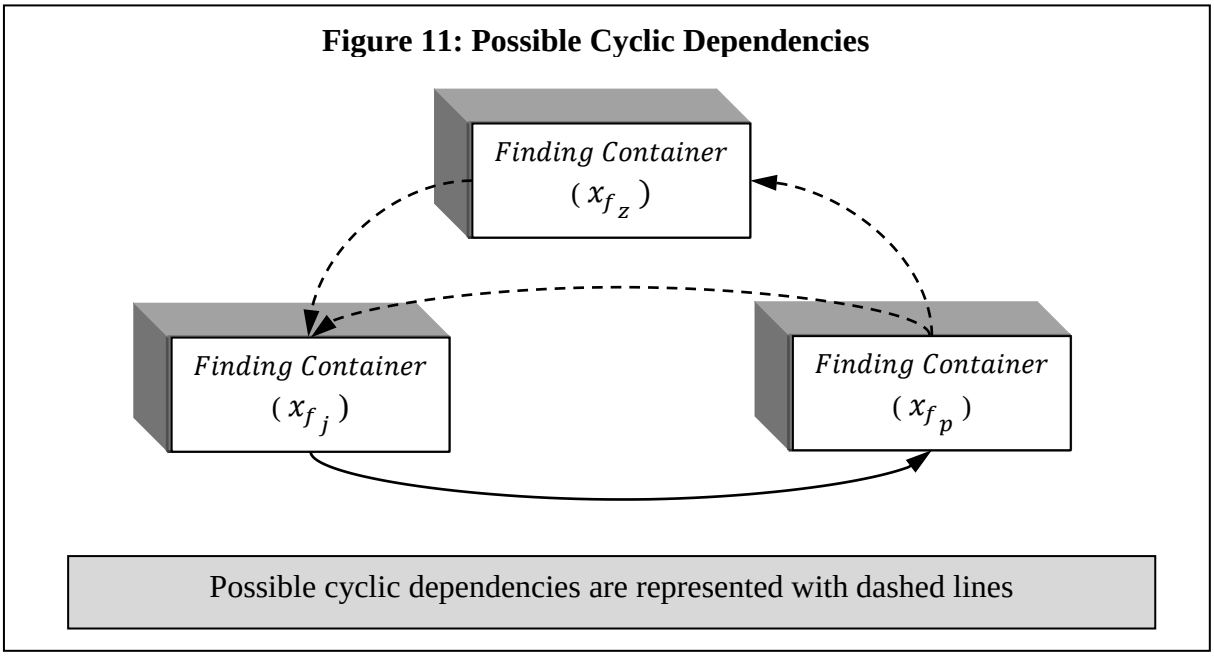
The Chain of Findings is a model that incorporates all the components and research up to this point and provides investigators with the necessary means to capture their reasoning process and all Finding Containers throughout the investigation. It incorporates the Time Stamping Authority (Chapter 3), Transformation System (Chapter 4) and Containers, as well as an interface to help investigators to use these components.

The Transformation System (Chapter 4) is used to create and transform Containers and ensure that any actions taken are recorded and captured within these Containers. The Transformation System uses the Time Stamping Authority in order to generate Proof of Actions for the actions that are performed by the investigator, providing accurate time information and non-repudiation. The Transformation System also configures these Containers with attributes using the attribute class repository in order to help circumvent syntactic and semantic differences among the attributes of the generated Containers. The use of such attributes is beneficial as they serve an important role in forming the building blocks of an investigator's arguments, namely $CACQs$. In Chapter 4 we discussed how the Transformation System incorporates Transformations for transforming Evidence and Finding Containers. We have already discussed Transformations regarding Evidence Containers and we now focus on Transformations regarding Finding Containers.

The Transformation System includes the necessary Transformations for investigators to access, view and modify Finding Containers, allowing an investigator to transform a Finding Container until it captures their reasoning process for a particular hypothesis. Investigators can also use the Transformations to import Finding Components from other Finding Containers, allowing them to reuse existing reasoning and to explore further hypotheses. Whenever an investigator imports Finding Components into new Finding Containers, these Finding Components are duplicated to avoid alterations to the original and the Finding Containers they may be a part of. This is done to help remove any problems with weighting schemes and weight conflicts, as well as update and deletion conflicts.

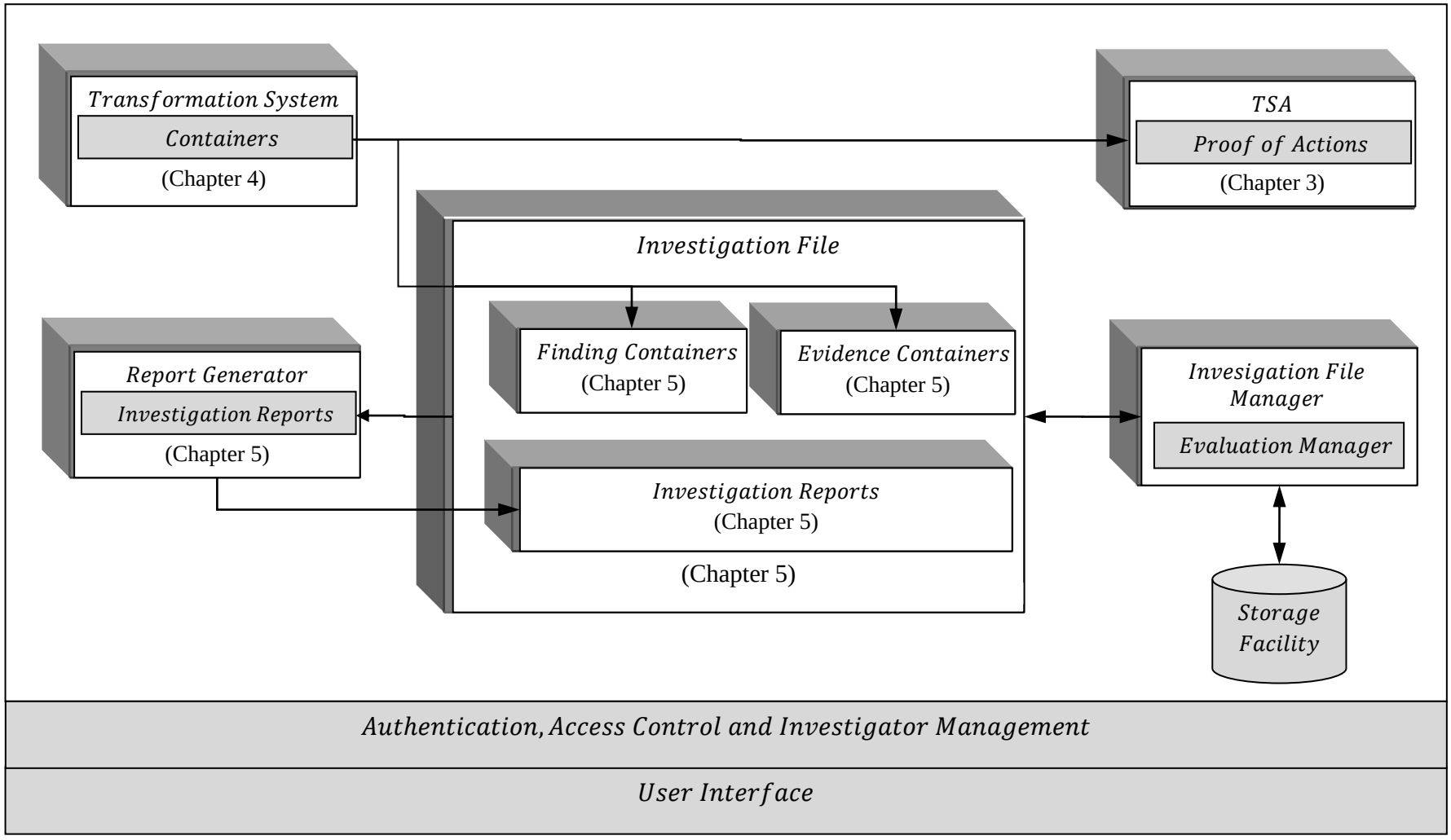
A particular concern with evaluating of these Finding Containers and their associated components is that of cyclic dependencies. A cycle can occur if a Finding Container (x_{f_j}) serves a particular role in another Finding Container's (x_{f_p}) Requirements, and x_{f_p} serves a role in x_{f_j} Requirements, or in another Finding Container that serves a role in the Requirements of x_{f_j} (demonstrated in Figure 11). By incorporating such a check within Transformations whenever an investigator attempts to commit a change to a Finding Container, it can help ensure that such a cycle is never committed. While the Transformation System provides the means to create and transform Containers, how does an investigator keep track of all the Containers for a particular investigation?

Figure 11: Possible Cyclic Dependencies



Each investigation may contain a large number of Containers (Finding and Evidence-based) as well as Finding Components. When capturing an investigator's reasoning process, we would like to allow all information regarding a particular investigation and an investigator's reasoning process to exist in a single particular location, making it more easily accessible. The Chain of Findings therefore incorporates a structure called the investigation file. In order to manage the creation, loading, and storing of these investigations files, the Chain of Findings incorporates an Investigation File Manager. The Investigation File Manager handles these tasks and ensures that the investigation file and all its contents are stored in the appropriate manner within the storage facility. By allowing only the Investigation File Manager to interact with the storage facility, all such logic is encapsulated within the Investigation File Manager, as well as all logic with regards to how the stored information is retrieved to construct the investigation file. In this manner, should the storage facility be changed, only the Investigation File Manager will have to be adjusted accordingly. The Investigation File Manager serves another important role.

Figure 12: The Chain of Findings



Each of the Finding Containers used to formulate and capture an investigator's reasoning process could consist of a large number of Finding Components. When evaluating a Finding Container to determine which conclusion to derive, each of these Finding Components must first be evaluated. These evaluations must be done efficiently, in order to provide relevant, timely and accurate information to the investigator. The Investigation File Manager contains the logic of the structure of the Investigation File as well as how the Investigation File and all its components are stored. Such detailed knowledge can ensure that evaluations are performed in the most optimal manner. The Investigation File Manager is therefore responsible for performing and managing these evaluations as well. The Investigation File Manager is also responsible for handling the executing of the Reasoning Algorithms for Finding Containers whenever necessary.

The user interface provides a single means for investigators to perform all their desired actions such as creating and accessing Containers, performing Transformations and editing their reasoning process. The user interface interacts with a number of components within the Chain of Findings, abstracting such complexity from the investigator and making the model easier to use. While the model has a number of uses and benefits, its user interface will play an important role in aiding investigators in using the model and its various components. Should the user interface require too many steps to perform certain tasks, or be too difficult to use — acting more as a hindrance — it may deter investigators. Manson et al. [42] discuss how despite Encase's complex and useful functionality that its poor design makes using it very difficult. They discuss how such poor usability may hence require more intensive training in order for an investigator to correctly use the software, increasing the costs and time for an organisation, and/or investigators to adopt such a tool. Designing an interface that is well-suited for its intended audience helps to ensure that the system is easier to use, easier to remember and that its users are shielded from the complexity of the underlying program [61]. We therefore will pursue further research to determine the optimal approach for creating such an interface for the Chain of Findings, allowing investigators to more easily use the model to achieve their goals.

Sharp et al. [61] for example, discuss a number of usability goals and design principles that should be taken into account when performing human computer interaction design. These usability goals and design principles, along with other forms of research such as surveys and case studies, should be used to design the interface so that it suits investigators' needs. Once the investigation is complete, or at periodic times during the investigation, investigators can use the interface to interact with the Report Generator in order to generate reports for their current progress in the investigation.

Our focus for this thesis was on capturing an investigator's reasoning process during an investigation; however, we also incorporate a means to generate reports, so that investigators can not only capture their reasoning process, but also present it. The manner and the detail with which an investigator presents their reasoning process will be based on the investigation, the intended audience, their technical understanding, and the approach that provides optimal clarity. Investigators should therefore have fine control over the layout of the reports, the report's aesthetics and what information should be included within the report. There may

exist a large number of Finding Containers and Evidence Containers, each containing a vast amount of information. Investigators should therefore be able to incorporate only relevant Containers, and specify the format that should be used to present this information, as well as the granularity thereof. In order to provide such behaviour, the Chain of Findings incorporates a Report Generator, and future research will investigate how to implement such a Generator.

In the Chain of Findings we also integrate an authentication, access control and investigator management layer. This layer is responsible for authenticating investigators and limiting the actions an investigator can perform during an investigation (based on skill, rank etc.), as well as performing investigator management. Investigator management is responsible for handling and storing all information regarding investigators, including their associated investigator IDs. It is also responsible for handling their public and private keys , which may be unique for each investigation for security reasons, and for storing the certificates binding investigators to their public keys. It is important to reiterate that an investigator's ID and their private key are used to generate the Proof of Actions and this layer is therefore responsible for making such information accessible to the Transformation Manager in a secure manner. While our dissertation does not focus on the aspects discussed in this layer, they play an important role in the Chain of Findings and further research is required to determine how to integrate them into the model.

5.8 Chapter Summary

In this chapter we introduced a new Container, based on the Evidence Container, which we call the Finding Container. Finding Containers allow an investigator to capture their hypotheses during an investigation and to capture and encode how they reasoned about them, whether they should be supported or refuted, and how they came to their conclusions. A Finding Container allows an investigator to use evidence (digital and physical evidence) as well as findings uniformly as building blocks for their reasoning. Finding Containers also allow investigators to capture and describe what impacts these building blocks have on a particular hypothesis and to describe how these building blocks are factored into their reasoning and the conclusions they draw. In addition, Finding Containers allow an investigator to factor certainty and alternate explanations into their reasoning, and to dynamically determine the impacts that changes may have upon their reasoning (so investigators can quickly get feedback). Finding Containers offer flexibility for investigators, providing them with the freedom to capture and formulate their reasoning. Furthermore, the capturing and the encoding of the investigator’s reasoning in a structured and non-volatile manner enables future investigators to build on their reasoning, and to better understand their reasoning process. After our discussion on the Finding Container we then discussed how Finding Containers play central roles in the Chain of Findings.

The Chain of Findings is the model that incorporates the various research and components introduced up to this point, providing a means for investigators to capture their reasoning process during an investigation as well as the steps they followed during their reasoning

process. The Chain of Findings also incorporates an investigation case file, a storage facility, a report generator and a user interface. The investigation case file allows investigators to store all Containers (Findings and Evidence) as well as all Finding Components that are relevant to the investigation, whether or not, they are associated with Finding Containers. In this manner, all of an investigator's building blocks and reasoning are located within a single location which is then stored in the storage facility. Finally, the user interface helps investigators to interact with all the components and helps them to more easily perform the tasks that they need to.

In this chapter, we discussed how the Chain of Findings and the contained Findings Containers meets the remaining requirements of our problem statement. These requirements are repeated for ease of reference. *The investigator must be able to capture their conclusions, the arguments behind their conclusions and the rationale behind each of their steps of their reasoning process. The Chain of Findings must provide flexibility in the manner in which investigators can formulate their arguments, factor in certainty and achieve their desired findings. The Chain of Findings must capture all details of an investigator's reasoning process, including the combination of building blocks that an investigator uses for their arguments. These building blocks may be evidence — digital or physical — and existing findings. Investigators must be able to use these building blocks uniformly and must capture the attributes and criteria that allow these building blocks to serve/not serve their roles in the investigator's reasoning. The Chain of Findings must also allow investigators to incorporate multiple confirmations for their reasoning, and it must allow investigators to capture alternate explanations and show why they cannot hold.*

In the next chapter we explore the uses of the Chain of Findings, its Findings Containers and other components in greater detail. We then provide an example of its usage to demonstrate the various aspects of the Chain of Findings, and how it can aid in investigations.

Chapter 6

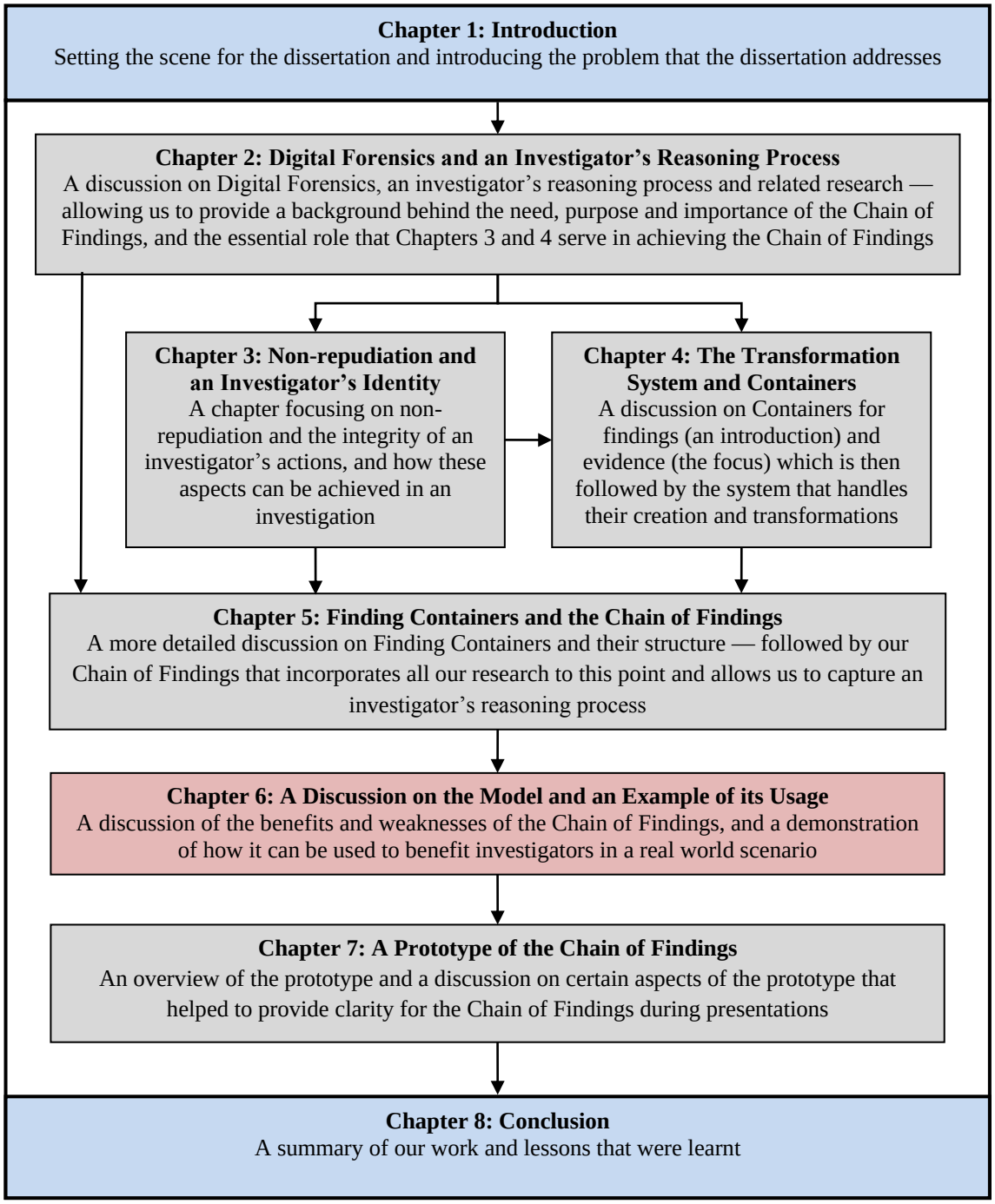
A Discussion on the Model and an Example of its Usage

Objectives

- To provide a deeper discussion on the model, its uses, benefits and drawbacks
- To provide an example of how the model can be used in a real world scenario and to discuss how it can benefit the investigators involved

SUCCESS is not final, failure is not fatal: It is the COURAGE to continue that counts
Winston Churchill

A Map of our Current Location in the Dissertation



Chapter 6: A Discussion on the Model and an Example of its Usage

6.1 Introduction

Organisations may be unlikely or unwilling to buy into new software, whether it's a new information system, a new database product or type [16] or a new application (suite). This is particularly so when an organisation already has software capable of doing similar tasks [15][29]. Despite the benefits that the new software may offer and its potential to aid the organisation, there are the costs of the new software and the costs of incorporating the new software into the organisation, the necessary training required so staff can use the software and the possibility for a reduction in productivity due to unfamiliarity with the software [16][29][61]. There are also concerns with the potential security of the new software and the potential vulnerabilities it may contain and/or introduce [52].

Our model will require an implementation to be of use to organisations and law enforcement. Organisations and law enforcement may already have existing approaches for performing and capturing their findings and their reasoning process. The Chain of Findings is a new approach and may be seen as complex and difficult to understand for investigators, particularly those who are new to the model. In this section we discuss why the Chain of Findings is beneficial to those who use it and provide a motivation for why implementation and research of the Chain of Findings should be pursued. We also discuss how it can aid in the vast number of growing incidents which may occur — criminal and civil, internal and external. Furthermore we elaborate on particular issues that must be taken into account by investigators when using the model.

Our discussion begins with, section 6.2, and it focuses on the detailed audit facilities of the Chain of Findings. In section 6.2 we discuss the detailed audit facilities that help to capture an investigator's reasoning process, allowing for a more thorough reconstruction. In section 6.3 we elaborate on how the Chain of Findings helps to ensure that investigators are always given feedback about their actions and feedback on how their actions impact their arguments. In this manner the Chain of Findings provides feedback on the investigators' progress in the investigation. We then discuss the flexibility of the Chain of Findings in section 6.4, and how it provides investigators with flexibility in their reasoning process. The Chain of Findings also provides flexibility in the sense that it can be adapted to cater for future forms of evidence and types of formats. The model provides further flexibility as it allows these new forms of evidence and formats to be treated uniformly. In section 6.5 we elaborate on an additional benefit, not yet discussed, for why the Chain of Findings should be pursued. Section 6.5 allows us to elaborate on how our model helps organisations and investigators to maximize their capability to collect credible digital evidence and minimize the costs of digital forensic investigations. This is known as Digital Forensic Readiness and it provides us with

an opportunity to discuss how the Chain of Findings can help ensure that organisations and investigators are better prepared. We then continue with how the Chain of Findings can serve as an important means for the training and testing of investigators and how the Chain of Findings can help provide guidelines for investigations. While Digital Forensic Readiness is not a requirement for our Chain of Findings, it demonstrates further benefits of the Chain of Findings as an encouragement for why such a model should be pursued and adopted by organisations and investigators. Section 6.6 then elaborates on how the model can be used in a real world scenario.

In the scenario we use physical and digital evidence to emphasize the model's capability to treat both types of evidence uniformly and we discuss how it can capture an investigator's reasoning process. The example therefore allows us to elaborate on the practical uses of the model and the various benefits it can offer to an investigation.

6.2 Detailed Audit Trail and the Reconstruction Capabilities

In Chapter 2 (section 2.3.5) we elaborated on a number of sources which highlighted the importance of audit trails and the importance of reconstruction. These sources include Rowlingson [58], the National Institute of Justice [47], Ahmad et al. [3] and the Association of Chief Police Officers [73]. Rowlingson discussed the concept of a case file, and how it is used to capture an investigator's hypotheses, the evidence that supports the hypotheses and the arguments showing that the evidence does confirm the hypothesis [58]. The National Institute of Justice discussed the examiner's report, and how it stores all the findings found during examination and analysis, how they were obtained and the processes and tools that were used in order to obtain them [47]. Ahmad et al. [3] emphasized the importance of auditing systems and their capability to capture relevant and correct information. The Association of Chief Police Officers [73] discussed how a detailed audit trail of all the steps followed, as well as the processes followed, should be recorded and properly preserved. The audit trail should be captured in such detail that a third party will be able to examine it and be able to reconstruct the process followed by the investigator and the investigator's results.

Many tools provide automated support for creating forensic reports that are based upon the investigator's notes and bookmarked information, and provide a form of a template that investigators can build on [37]. An investigator's notes, and the reports they produce, may however fall victim to fatigue, laziness, forgetfulness, human error and a lack of detail — hence, the forensic reports based on the notes will be affected as well. An investigator's notes and their reports, particularly when compared to another investigator's, may furthermore lack consistency in terms of detail, terminology and format used. Should these notes and reports be done post-investigation, certain information may be forgotten, lost or simply omitted, reducing the notes' and reports' potential value in confirming the soundness of the forensic process an investigator followed. This is particularly so due to the large periods of time over which investigations may draw. Overall the interpretation of an investigator's reports and notes may suffer as a result, making it a difficult and time-consuming task.

The Chain of Findings ensures that all relevant information regarding an investigator's reasoning process is captured and stored accurately, in detail, in a standard, persistent manner as it occurs. By doing so, the Chain of Findings allows an investigator's train of thoughts and actions to be encoded and captured. The captured information provides a type of explanation system, similar to that of an expert system, for an investigator's reasoning process and how they reached their conclusions. The Chain of Findings however requires input from the investigator to do so, and such manual input can reduce productivity in an investigation. The capturing process allows investigators to capture their train of thoughts as it occurs and provides the investigator with structures to capture their reasoning, as well as to present it. The user interface should also be designed to allow investigators to perform such tasks as easily as possible, so that the capturing process does not become a hindrance to the investigation. The Chain of Findings helps to automate the capturing of important information during an investigator's reasoning process and offers a number of benefits.

Our intention was to capture the reasoning process of an investigator in a well-defined manner so that a detailed record of their actions and events would be sufficiently captured, should integrity or detailed information be challenged. In our background we noted how investigators' actions can affect the conclusions they derive [73], their interpretation of the evidence and their overall reasoning. An investigators must therefore take sufficient precautions during the investigation where these precautions extend to the tools they use. Investigations can affect a suspect's life, their job and whether disciplinary action is to be taken — and in more severe cases whether criminal prosecution should take place. Capturing an investigator's actions in detail provides a means to verify the investigator's claims and findings and provides a more sound manner for recording the steps they followed. This supports the forensic soundness of their arguments, their objectivity and their reliability [58]. It also offers a means for a third party to reconstruct and challenge their results and reasoning [47]. The capability to challenge and reconstruct the process that the investigator followed provides a form of verification for an investigator's reasoning. It also helps to avoid biased or inaccurate conclusions and arguments as they can be more easily challenged. The model uses Proof of Actions to strength the reliability of the captured information.

Proof of Actions help to provide reliability to the captured information through the use of digital signatures and timestamps. The use of digital signatures helps to ensure that an investigator is accountable for their actions, and that a third party can easily determine which investigator is responsible for performing certain actions. The timestamps are generated using a Time Stamping Authority, which helps to ensure that all timestamps are consistent throughout the investigation in terms of accuracy and format. In this manner the timestamps help to provide a more detailed recollection of an investigator's sequence of actions and the times that they occurred. In doing so we are able to construct a more reliable timeline of the investigator's actions, providing a stronger form of integrity for the process that was followed during the investigation. The timeline and the associated timestamps aid in the reconstruction of the investigators actions, and the sequence they followed should it be challenged in cross-examination, or by an opposing party, strengthening the reliability of an investigator's results.

Another particular aspect that relies strongly on accountability and the time that events occurred, is the Chain of Custody.

The Chain of Custody is an important aspect of an investigation as it allows one to easily determine where evidence is at a particular point in time and who is accountable for it (section 3.2). The Chain of Custody provides a form of non-repudiation, provided sufficient access control systems are used and other controls are in place to prevent modifications and fabrications to the Chain of Custody. The Chain of Custody plays a vital role in an investigation as it ensures that, throughout the period in which the investigative team had the evidence, sufficient care was taken to protect the integrity of the evidence. It also plays an important role in expert testimony as a way to argue about the correctness of evidence. The model provides thorough provenance information through the use of Transformations and Proof of Actions.

Once an investigator uses a particular Transformation to transform a Container, such as an Evidence Container, the particular Transformation has the capability to alter and/or spoil the Container's contents. The Transformations hence capture detailed information with regards to the Transformation test history, preconditions, post conditions, Transformation details and Transformation operations within the Container's Proof of Action History. The recording of such information shows that the Container's integrity and its contents are preserved throughout the Transformation as well as the reliability of the Transformation itself. In addition, recording the operations of a Transformation aids the reconstruction of a particular Container and how it came to be, further supporting the reliability of the Container.

In this manner all information regarding how these Containers came to be in the required form for them to serve as building blocks in the investigator's reasoning, is easily contained in a single location, namely the Container itself. Information regarding the Container is not distributed across multiple sources, such as the Chain of Custody, paper logs, test documentation, investigative notes and forensic reports. Furthermore, these Containers provide the means to keep track of the history of Transformations that have been performed on them along with any other provenance events, such as transportation and collection. By doing so, these Containers provide a means for capturing provenance events regarding digital and physical events. These events are recorded using Proof of Actions to ensure that they use accurate times for when events took place and to provide non-repudiation for the events that occurred.

Finding Containers encapsulate an investigator's arguments for why their hypotheses are supported or refuted, along with their reasoning and the rationale behind these arguments. By doing so they encapsulate how the investigator's arguments are structured, and the building blocks that form this structure — whether these building blocks are Evidence Containers and/or other Finding Containers. Finding Containers also incorporate various description elements so that they can capture the reasoning and rationale of these arguments, as well as how they are structured. Should investigations then draw over large periods of time and the investigator responsible leaves the particular case, all this information is then available and maintained so that the investigation can commence without resources being wasted on

investigative work that has already been completed. Finding Containers further aid in this regard by encapsulating the Containers used to form an investigator's arguments as well as the relationships between them.

Corroborating Containers are uniformly captured in the Requirements of a Finding Container. The manner in which these Requirements are combined, namely using logical operators, allows the investigator to specify the complex relationships between these Containers. In order to capture more in-depth information about how the Containers support these Requirements, the model relies on $CACQs$. These $CACQs$ capture the attributes of the Container, the criteria that they (must) comply with and a detailed description of why such criteria are important, in order for the Container to support the particular Requirement. The investigator then combines these Requirements using logical operators to form the confirmations for their Supporting and Refuting Arguments). In the next section we discuss how the Chain of Findings helps to provide feedback to an investigator about their progress in an investigation, but before that we present the Chain of Findings as a metaphor for a game.

6.3 Progress and Certainty Feedback

McGonigal [44] explains how the four most defining traits of a game are a main goal, a set of rules to comply with, voluntary participation and feedback. The Chain of Findings can be seen as a form of a game, namely where the main goal is the particular hypothesis, assuming a goal-driven approach, that the investigator would like to achieve or the particular conclusion the investigator would like to deduce at the end of the investigation. Allowing an investigator to specify a main goal for their investigation, such as whether or not the evidence supports party X's involvement in an incident, helps ensure that the investigation remains focused. This is particularly useful in lengthy investigations. The rules are the particular legal requirements that an investigator must adhere to, in order to ensure that their evidence and reasoning are logical and forensically sound, as well as that their steps can be reconstructed to verify their results.

Voluntary participation does not necessarily apply to an investigator being able to leave and join an investigation whenever they please, but rather the fact that our model allows an investigator to work on multiple aspects of their reasoning as they wish. In this manner the model allows an investigator to create and work on multiple sub-goals and to switch between them as needed. Should an investigator then encounter a dead end, they can easily pursue other avenues of interest, while still remaining focused. Investigators are also free to choose which evidence to work on at which particular time, and to relate them to various Requirements as they are able to — emphasizing the non-linear capabilities of the model. Regardless of the approach chosen by the investigator, the model continuously provides feedback to the investigator with regards to their progress.

Feedback is the process of sending back information to the user about the actions they have performed [61]. Feedback allows the user to not only see the impact of their actions, but it also demonstrates to the user how their actions are helping to achieve their intended goal(s)

[44]. The model allows an investigator to specify the required certainty threshold for particular aspects of their arguments. The use of feedback in the model, coupled with certainty weights and measurements, allows the investigator to see their progress, and how their actions and analysis are helping to achieve the investigator's required certainty, and hence their goals. Feedback therefore aids the investigator by providing information about the certainty with which they can make certain claims and allows them to be better prepared for possible challenges when presenting their expert testimony in the court of law. Feedback in this sense helps to motivate [44][61] the investigator, especially in scenarios where the investigation may be complex and time-consuming**.** Progress feedback is also beneficial in scenarios where organisations must determine if further resources — monetary, expertise and time — must be allocated to an investigation.

An investigator must however take care when choosing certainty weighting algorithms, certainty weights, and the required certainty thresholds. While they can be based on expert knowledge, they will typically be subjective. Templates derived from existing incidents and investigations as well as training (section 6.5.1) can however help in this regard. They can help to ensure more accurate weight assignments and that thresholds and weighting algorithms are better chosen. While the model provides flexibility in the weighting schemes an investigator can choose and build on, it also provides flexibility in its capability to grow and adapt to meet future forms of evidence and format types.

6.4 Flexibility and Adaptive Capabilities

Gamma et al. [26] and Turner [68] discuss how, when one designs a model, it should be designed to be as flexible as possible (Chapter 4). The model should take into account current problems and concerns and it should also cater for future ones. In software design the importance of flexible design is emphasized as software solutions can be easily adapted to cater for future needs, problems, and requirements without requiring an expensive overall redesign [26][46][48]. Flexibility also plays an important role in problem-solving mechanisms, such as in the Artificial Intelligence domain, as it allows one to attempt to solve a problem using a combination of approaches [38] and to find the most optimal manner for doing so [18]. Epp [25] discusses the benefit of such flexibility when he discusses how different approaches may be better suited for solving specific problems.

The National Institute of Justice [47] discusses how every case is unique and hence contains its own set of challenges that need to be addressed. Each of these challenges may require different approaches in order to ensure that they are sufficiently resolved. In our dissertation the particular problem an investigator attempts to solve is a hypothesis they would like to confirm or deny (the goal) or, given a collection of evidence, what conclusions (the goals) they can they derive. Sharp et al. [61] discuss how, when one designs systems, one should take into account what people are good at. They discuss how people are not purely linear beings that follow a single approach when they solve problems. The model hence provides investigators with the freedom to pursue data-driven approaches, goal-driven approaches and a combination thereof when formulating their arguments in an investigation. In this manner an investigator is given flexibility in the approach they decide to use to best address the

investigation. Data-driven approaches allow an investigator to use existing Evidence Containers and Finding Containers in order to begin formulating conclusions. Goal-driven approaches allow an investigator to decompose the hypothesis they want to confirm or refute into smaller, more easily achievable goals, such as into the Requirements that need to be met. In this manner the investigator can decompose complex hypotheses into more easily attainable steps, using a combination of goal-driven and data-driven approaches to achieve their desired goals.

Investigators are also given flexibility in the sense that they can not only confirm a hypothesis (or derive a particular conclusion) with a certain level of certainty but also show that it's not possible to deny it (with a certain level of certainty). This form of flexibility allows an investigator to be better prepared for cross-examinations when presenting their results. This is due to their capability to argue a particular hypothesis or conclusion from both sides (supporting and refuting), showing that they have taken into account inculpatory evidence, exculpatory evidence and evidence of tampering and that their arguments are sound and can be relied upon. Investigators may also use a similar approach to show that a particular hypothesis cannot hold with a certain level of certainty, providing them with the flexibility to argue using an approach that best suits their needs. Furthermore, investigators are given the freedom to choose weighting algorithms that best suit their needs and intentions, as well as the choice to use no certainty weighting algorithm or weights should they want the model to rather just capture what they do. In addition our model incorporates flexibility in that it is able to expand according to future needs.

There is an increasing number of computer-related incidents, expanding in diversity, severity and complexity [4][47]. This increase means that we must take into account not only today's incidents but future ones as well and that the model must be able to adapt to meet these future needs. It also means that the model should be able to easily adapt to handling new forms of evidence and the Transformations that this evidence may require, so that investigators can easily handle future scenarios. The Transformation Manager incorporates the capability to register new Transformations to cater for an investigator's needs and to cater for new forms of evidence — including new sources of evidence and new storage formats. In addition, the Chain of Findings allows an investigator to treat evidence, digital and physical, along with existing findings uniformly and to combine them as they see fit to formulate new findings. In this manner, an investigator is given greater flexibility in how they go about formulating further findings. To capture such reasoning, a Finding Container is reliant on attributes of the Container through the use of the $CACQs$ and their contained $ACQs$.

$CACQs$ allow an investigator to not only capture the Container of interest, but also which attributes of the Container and the reasons behind why these attributes serve/do not serve their role in the investigator's reasoning. These $CACQs$ allow an investigator to treat all Containers uniformly, whether they are evidence-based (digital or physical) or finding-based, through the $CACQs$ declarative nature. These $CACQs$ and their associated $ACQs$ also provide data independence, as the investigator can treat these Containers uniformly. In order for such a structure to serve its role in the Chain of Findings, it is however strongly reliant on these attributes and hence the attribute class repository from which these attributes are created.

The attribute class repository allows for new sources of evidence, as well as new formats for storage media, to be catered for, as new attributes can be added to it when needed. The main problem with the use of such an attribute class repository is that, should an attribute need to be changed — perhaps its name — its description, representation or its range of possible values — it can affect all Containers that are configured with such an attribute. There may also be conflicts if an attribute's semantic meaning is changed, affecting the Containers which are configured with such an attribute. The attribute class repository should therefore be well-established, well-researched and well-accepted in the Digital Forensics community in order to help overcome such problems. We now explore how the Transformation System and the Chain of Findings can help to provide an additional benefit for organisations and investigators, namely in terms of Digital Forensic Readiness.

6.5 Digital Forensic Readiness

Rowlingson [58], discusses a number of steps for helping an organization achieve Digital Forensic Readiness. Before we elaborate on his research and how our model helps to aid in Digital Forensic Readiness we must first discuss what Digital Forensic Readiness is. Digital Forensic Readiness is when an organization has all the necessary means and infrastructures in place in order to collect and process digital evidence, should an incident occur [64]. The means and infrastructures include security policies, procedures, practices, mechanisms and training programs [24] and they allow for a proactive approach to an investigation. The proactive approach aims to minimize the cost, labour, time and resources required for an investigation, especially when compared to more reactive approaches. Following a proactive approach helps to reduce interruption to the organization's operations [58] should an investigation occur. It also helps to maximize the capability of an organization to have more sound approaches for capturing, processing and handling digital evidence [24][58][64].

Digital Forensic Readiness should aid in accelerating a forensics investigation and to reduce the costs of performing such an investigation [64]. Endicott-Popovsky et al. [24] discuss how in certain situations, time will directly impact cost, as after an incident occurs, important systems may need to be restored to a secure state before they can go live again. They also discuss how when time is of the essence, sufficient means must be in place to ensure that an investigator can collect credible digital evidence from these systems in a forensically sound manner in a reasonable period of time.

One must define realistic business scenarios (and incidents) which may require digital evidence [58]. Doing so allows an organisation to perform risk assessment and vulnerability assessment. When an organisation follows such a step, it can assess potential internal incidents, potential external incidents and potential legal lawsuits and, overall, determine likely situations in which evidence may benefit the organization. This step encourages the organisation to think about the training required by their staff, the evidence that may be required and useful in these particular scenarios, as well as the best manner to capture the evidence in a forensically sound manner. It also encourages an organisation to think about the various policies and procedures that need to be in place to avoid privacy and confidentiality

conflicts from occurring when collecting the evidence [58], should these particular scenarios/incidents occur. This step therefore helps an organisation to begin preparing and planning for when such a scenario or an incident occurs and forensic action needs to be taken [64].

The Chain of Findings allows one to develop an argument towards a particular hypothesis (and for making certain conclusions), showing not only which Containers support or deny such a hypothesis, but also how they aid in doing so. These arguments provide a detailed step by step discussion on how one's arguments are formed, and can also be seen as a guide for the reconstruction of one's arguments and reasoning process. Should one create well-formed Finding Containers for organizations that mimic real-world scenarios, using realistic Containers, it would enable the template to capture how to formulate well established, logical and legally sound arguments should a particular incident occur. In this manner the Chain of Findings will allow organizations and investigators to learn from these templates. These templates can serve as a guideline offering a means for an investigator (and organizations) to learn how to avoid making poorly supported arguments, and common pitfalls in their arguments with regard to these scenarios/incidents. These pitfalls include conclusions which do not derive from the facts [50]. The template can therefore help specify the steps an investigator must follow in order to support or refute a hypothesis in a legally sound manner. Examining and using these well-formed templates can help organizations to ensure that the necessary measures are in place should these incidents occur. In addition, the templates can help investigators to better allocate their resources, particularly when an investigator must use multiple confirmations and multiple refutations.

In section 2.4.2.2 we discussed how an investigator must use multiple confirmations to help support a hypothesis and multiple refutations in order to help refute a hypothesis. Cohen [15] discusses how legal matters may be inconsistent with adequate logical confirmations or refutations of hypotheses and hence encourages an investigator to use multiple confirmations and refutations to circumvent such outcomes. The Chain of Findings allows for an investigator to use a number of $WeightedOrRequirements$ to capture these multiple confirmations and multiple refutations. Cohen also discusses the various resource constraints in an investigation and the limits that they impose on what can and cannot be done in an investigation. Cohen hence emphasizes the need for direction in an investigation. Templates help to provide direction through the guidelines that they specify and, in doing so, they help to ensure that resources can be more efficiently applied to an investigation. The guidelines also help to provide direction by specifying the potential types and sources of evidence that may be useful in such an incident (benefitting the associated pre-search phase).

Rowlingson [58] discusses the importance of identifying potential types and sources of digital evidence so that sufficient preparations can be performed and precautions taken. The Department of Justice [2] discusses how different scenarios and incidents will require different forms of evidence. The selection of business scenarios that apply to an organization can help ensure that the organisation caters for the necessary evidence, in terms of audit facilities and in terms of preparations and precautions. Investigators can also benefit from such information as it can help them to narrow down potential sources should, a particular

incident occur, and to ensure that these potential sources are collected. These preparations and precautions are necessary to ensure that sufficient means are in place to collect, and protect and preserve the relevant information, and its integrity [64] — avoiding damage and spoliation of the collected information [73]. An organization may have a number of computing devices and systems including cell phones, fax machines, information systems and complex networks. The vast number of potential sources of digital evidence, as well as physical evidence, makes the identification of these sources, which may be vital in these particular scenarios, difficult and time-consuming should, no guidelines be available.

The Chain of Findings templates help identify and provide information about these potential sources by capturing and describing what (types of) evidence is (are) required in order to make certain claims or refute them, aiding organizations and investigators involved in the investigation. The templates also capture information on these sources' potential to serve as inculpatory evidence, exculpatory evidence and evidence of tampering, and how they can be incorporated into an investigator's reasoning, to ensure it is more sound. The use of such information helps to identify the potential roles that certain evidence can play in that particular scenario/incident. Organizations can then use this information of evidence sources contained in the Finding Container templates in order to ensure that they are sufficiently prepared should such an incident occur. In order to ensure that an organization is properly prepared and able to collect and examine evidence, it must also have the necessary tools in place. Investigators must also have the necessary training to use these tools.

Organizations and investigators must be sufficiently prepared for the new and ever-growing number of incidents, as well as the growing number and forms of digital evidence [47][73]. The templates capture and describe what Transformations may need to be performed on evidence in order for it to be transformed into the correct form in these particular incidents, should realistic digital Evidence Containers be used. Capturing such information helps organizations to further determine the skills and training that will be needed by an investigator in order to carry out these Transformations and reasoning. Investigators may also use the list of Transformations described in the templates in order to ensure their Transformation Manager is able to perform the necessary Transformations and that these Transformations are up to date. Furthermore, an investigator may use the templates and the associated realistic template Containers as a reference to confirm that the Transformations are working correctly. The templates can therefore help ensure that investigators and organizations are able to perform the necessary Transformations, particularly if realistic Evidence Containers are used.

By using realistic digital Evidence Containers which mimic realistic and potential evidence that an investigator may encounter in the templates, and by designing templates to mimic real-world incidents, organizations and investigators can be better prepared should such incidents occur. Preparation plays an important role in forensic analysis, incident response and evidence collection. The more realistic the templates and the Containers and, hence, the more realistic the provenance information in the Container, the more realistic training using the templates can be. This helps to ensure that investigators and staff can receive better real-world experience from the templates and associated training [27]. Sufficient care must

however be taken into account when constructing such realistic evidence [45] Containers and templates, to avoid any legal issues such as privacy concerns and confidentiality violations. This is particularly so should the templates and Evidence Containers be derived from real world incidents. While realistic Evidence Containers will most certainly be useful in these templates and provide detailed guidelines for an investigator's reasoning process and preparation, they may not always be feasible or available, and hence cannot be relied upon. The use of Null Containers or example Containers will still allow the templates to mimic real-world incidents, to describe the relationships between evidence, to describe the potential types and sources of evidence and to emphasize the importance of the evidence's attributes when an investigator forms their reasoning. Detailed provenance information for these Containers and the Transformations required may however not be easily available, but guidelines for the reasoning process will be. In this manner the templates can still aid in forensic readiness, whether or not realistic Containers are available.

Should a particular incident occur, an investigator can then load the relevant template and begin following the steps and guidelines dictated in the Finding Container template, provided that the necessary preparations and training have been completed and are in place. Having a detailed template, and its inherent, well-formed structure, helps the investigator to quickly and easily perform the investigation with regards to the incident (or investigating the hypothesis). These templates and their detailed steps can therefore help to guide the investigation, aiding it to be done more rapidly, thoroughly and correctly — with minimal effort and disruption to an organization, their business and their productivity. Depending on how closely the templates mimic the real world incident and the investigator's particular investigative case, these templates may be used more than just to provide a guideline, and instead serve more along the lines of a 'fill in the blanks'. In this manner they can help to speed up the forensic process, even for new incidents. Overall, Finding Container templates help to reduce the cost, labour and time delays caused by an investigation [58].

Organizations and investigators can then continue reaping these benefits and keep up with the growing number of incidents that are relevant to them by ensuring their Chain of Findings templates are up to date through an appropriate update mechanism. Organizations and investigators can then use these new and updated templates to ensure that they are sufficiently prepared should the incidents occur. We now discuss how the templates and their aid to Digital Forensic Readiness can benefit organisations and investigators in terms of clients, automating the reasoning process and in terms of the training and testing (section 6.5.1). In section 6.5.2 we then discuss how the templates can benefit an existing Digital Forensic Readiness project, namely the CTOSE Project, and elaborate on how it could benefit future projects as well.

6.5.1 Clients, Automation and the Testing of Investigators

Chain of Findings templates can be used for three other main reasons with regards to Digital Forensic Readiness — namely generating trust, providing a means to test investigators and the automation of an investigator's reasoning process. Organizations can use these templates and their associated preparations to demonstrate to their customers that sufficient means are

in place to protect against these incidents and that, should they happen, that the organization has the necessary means in place to react quickly and efficiently. By doing so it can help ensure that customers feels safe with the organization and trust the organization, as the organisation has the necessary means in place to repel, recognize, recover from these attack and pursue the attackers should such events occur [24]. These templates and their associated preparations may also be used to attract potential customers, particularly those who are security conscious. Templates can aid in the efficiency of an investigation and by enabling them to aid in the automation of the reasoning process they could help further speed up investigations.

The templates could also provide a means for automating a large portion of the reasoning process. These templates can be used as a means to train systems, such as neural networks, so that they can learn from these templates. By doing so they can learn which Containers may possibly be related, how to better associate weights (benefitting Bayesian Networks) as well as how and why to form arguments. The well-defined, uniform structure used for the Finding Containers (and hence the templates) and the use of consistent and explicitly defined attributes to overcome syntactic and semantic heterogeneity can help ensure that training can be done more effectively. Description elements may require some conversion before they can be easily interpreted for training systems. In addition, the templates may be used as guidelines for Transformations, along with known file filters such as that in FTK (Chapter 2). In this manner they can aid Transformations in quickly narrowing down which potential evidence to search for, helping to speed up the process and to provide more efficient means for data reduction — especially where there are large volumes of data. In this manner templates can help to drastically reduce the large volume of data that an investigator is required to analyse. Templates can also provide a means to test investigators and verify their capabilities.

Many investigators lack the necessary technical and legal skills and training to perform a sound digital forensic investigation, resulting in a number of mistakes in the investigation [9]. Broucek and Turner [9] go on to state how many investigations suffer and fail due to these mistakes. Broucek and Turner hence emphasize the need for a formal, professional accreditation for investigators to help ensure that that they are sufficiently prepared, skilled and trained for investigations. Organizations may use these templates as a means to test and train investigators' reasoning capabilities, and not only access their capabilities but provide a form of a standard for grading investigators and their capability to handle certain scenarios (and incidents). In this manner it provides an organization with a means to verify their investigators' capabilities to handle particular incidents. Templates also help to train investigators on how to formulate arguments objectively, how to better associate weights, and how to reason objectively. Templates can also be used to test these aspects. Should an organisation lack the necessary skills to perform a particular investigation or phase thereof, it may require the help of external consulting agencies (Chapter 2). The templates can then help provide a means to verify the capabilities of these external consulting agencies. While future research is required in order to so, this section helps show that the Chain of Findings can

offer a number of benefits to organizations and investigators, and to Digital Forensics Readiness.

6.5.2 The CTOSE Project and Finding Container Templates

The European Union project Cyber Tools On-Line Search for Evidence (CTOSE) [8][9] has developed a methodology which is intended to aid organisations in performing sound forensic investigations. The project was began due to the fact that in most organisations, IT security management mainly focuses on Digital forensics (lowercase 'f') namely resistance, recognition and recovery, rather than Digital Forensics (section 2.2), which incorporates redress. Other reasons include how, when investigations take place there is often a lack of experience and knowledge of how to proceed in an investigation both technically and legally. Hannan et al. continue with how organisations are reluctant to follow full-scale investigations due to their potential costs and how they would rather recover from the incident's effects, potentially losing valuable attribution information as a result [8].

The CTOSE project intends to overcome these problems and incorporates a reference process model, a detailed examination of requirements, and a CTOSE demonstrator [8][9] to help ensure that investigators are better prepared for investigations and that organisations are forensically ready. The reference process model resembles guidelines at the technical, organisational and legal level, for how to ensure that digital evidence is admissible. The model describes the potential actions and decisions that an investigator may have to take during an investigation, helping to ensure they are better prepared. The model also includes detailed checklists that pertain to what information may be relevant in an investigation, allowing first responders to better preserve this information. The reference process model is then linked to a detailed examination of requirements that should be met at the technical, organisational and legal levels of an investigation, which is then linked to a CTOSE demonstrator. The CTOSE demonstrator then uses the reference process model and the Requirements to walk users through a variety of scenarios, educating them and providing valuable insight into the handling of evidence throughout the investigation. The CTOSE project therefore can help investigations to occur more quickly and soundly from both a technical and legal perspective.

The use of Finding Container templates can benefit projects such as the CTOSE project by providing more in-depth demonstrations and guidelines for how an investigator can use the evidence to formulate arguments in particular scenarios. The template can help provide more in-depth information about the evidence, the potential types of inculpatory evidence, exculpatory evidence and evidence of tampering that should be searched for in these scenarios and how they can be used to aid an investigator in their reasoning process. It can help demonstrate the potential relationships between evidence and it can help to better educate investigators on how to sufficiently support/refute their arguments. The templates also provide information on the attributes of the Evidence Containers, and how they are useful when forming these relationships, providing more in-depth guidelines to investigators. In this manner the Finding Container templates can be incorporated into the CTOSE project, so that not only is a checklist of evidence provided but the reasons of why the evidence is

useful in the investigation can be specified as well. In this manner the Chain of Finding and the Finding Container can possibly benefit further projects by helping to provide more in-depth information on the reasoning process. The incorporation of Transformation information into these templates can help further aid projects by providing Transformations guidelines. Now that we have discussed how organisations can use the Chain of Findings to be more forensically ready, let us explore how the model can be used to aid an investigation with the use of an example.

6.6 An Example of the Chain of Findings in Use

In order to convey how the Chain of Findings can be used and how it can benefit investigators and investigations, we make use of a particular fictional scenario. The scenario allows us to emphasize and discuss how the model can be used by an investigator to incorporate physical and digital evidence into their reasoning. The scenario allows us to describe how an investigator's reasoning is captured as well as the certainty that they place in their reasoning. In the example we elaborate on how $CACQs$ can be used and how Null Containers can be used in an investigator's arguments, as well as the benefits that they offer. In addition, the example allows us to demonstrate how the Chain of Findings aids in capturing the relationships between Containers through the use of Finding Containers.

The example helps to demonstrate how Finding Containers can become quite lengthy and how, through their capability to be expanded and collapsed, information can be shown at the desired granularity. In order to make the example more easily understandable and concise, we have however only included Supporting Arguments and description elements for the $CACQs$ and the $ACQs$. The Supporting Arguments allow us to discuss how a Finding Container can be created and used while reducing the overall size of the example. The use of only a subset of the specified description elements allows us to demonstrate their importance, while ensuring that the example does not become too long. We also use only UTC time in the example.

The layout for our example comprises sections 6.6.1 to section 6.6.3. In section 6.6.2 we discuss the particular fictional scenario that will be used to demonstrate the Chain of Findings' capabilities. In section 6.6.2 we discuss how the investigator then uses the Chain of Findings to formulate and capture their reasoning process in the investigation. Finally in section 6.6.3 we provide a discussion on the usage of the Chain of Findings in the particular scenario.

6.6.1 Scenario

In our fictional scenario we have a cellular company called Fodacom, and an employee, Frank Furt, that works at one of Fodacom's Pretoria branches. In this particular scenario a client's, Maria Madam's, SIM card was SIM swapped on 14 May 2012 at 10:45 (UTC) without her permission. Large sums of money were then removed from her account amounting to R100 000, according to bank statements through the use of the SIM swap. Fodacom's user log records show that Frank Furt was the party responsible for the SIM swap. The log records show that the login came from Frank's workstation, through the

identification of the recorded static IP and the recorded MAC address. It is important to note that all Fodacom workstations have a static IP. Based on the log records, Fodacom would like to take legal action against Frank Furt for his involvement in the SIM swap. Frank Furt however argues that he was at lunch during this period, from 10:00 to 11:00 and states that he had no involvement in this transaction, and is hence innocent. Fodacom requests the assistance of a Digital Forensic investigator to assist in the matter.

Fodacom restricts access to a large number of possible evidence sources from the investigator, including email accounts of its employees, due to the investigator not being an employee of the organisation. Fodacom does this due to the large amount of confidential information they contain with regards to employees and clients. Further reasons include the potential for decreases in productivity, and damage to their public image, should they provide greater access and a larger investigation be required. Fodacom hopes that by restricting the evidence sources available to the investigator, that the incident can be seen as a minor incident and not affect their clients' trust in the company. The particular pieces of evidence collected and that are made available to the investigator are:

- CCTV footage of the cellular branch on 14 May 2012
- Frank Furt's workstation from the branch that he uses in order to perform his daily tasks for Fodacom. The workstation uses Windows 7, 32bit operating system and is given to the investigator in a powered off state.
- User log records of all employees that logged in and out of the Fodacom network on the 14th of May.

6.6.2 Using the Chain of Findings

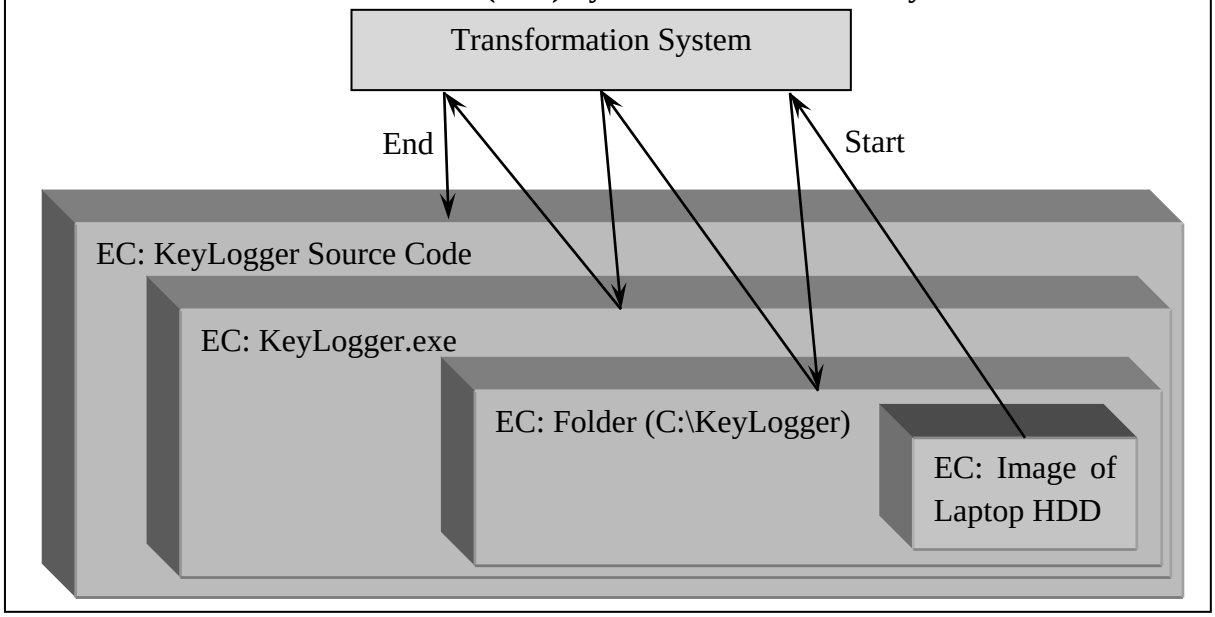
The first step the investigator takes is with regard to the suspect's workstation, and it involves imaging using the Transformation System and fingerprint analysis. The investigator uses the Transformation System in order to create a Container for the physical hard drive of the suspect's workstation, allowing the investigator to capture provenance information and properties of the device. The investigator then uses the Transformation System in order to image the hard drive of the suspect's workstation and to store the image into an Evidence Container. The Transformation System then configures the Container with the appropriate attributes of the hard drive and the associated image, as well as its hash. The workstation tower, screen and keyboard and mouse are then sent to a forensics laboratory for fingerprint analysis to determine which parties may have been in contact with the workstation. The results state that two sets of fingerprints were discovered, the first set of fingerprints being that of Frank Furt and the second set belonging to a fellow employee, Dave Culprit. These two sets of fingerprints are then stored in two separate Evidence Containers through the use of the Transformation System. The investigator then commences with the examination of the image of the hard drive of the work station.

The investigator uses the Transformation System to view the contents of the image, of which all details of the Transformation are stored in the Container. The investigator realises that there are a large number of files, even after the reduction of known files. The investigator, having worked on a similar case where a key logger compromised an innocent party's

password, navigates to the Startup folder (C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup) in the image. In the Startup folder, the investigator discovers a particular shortcut to an application named KeyLogger.exe. The investigator requests that the entire Startup folder be extracted into an Evidence Container. The investigator also requests that the Transformation System then extract the KeyLogger.exe shortcut into another Container. In this manner the investigator can more easily reason about the attributes of the shortcut, and use these attributes in their Finding Containers.

The investigator views the time of creation attribute, the time of last modification attribute and the shortcut's target location. The investigator discovers that the time of creation and the time of last modification are both equal to 08:00 02 February 2012, which is before the incident occurred. The investigator then examines the shortcut's target location and discovers a folder in which the actual executable resides, namely C:\KeyLogger. This particular folder is then extracted into an Evidence Container, from which the investigator asks the Transformation system to extract the particular KeyLogger.exe into a Evidence Container as well. The investigator then uses the Transformation system to decompile the executable, and returns the source code of the file into a new Evidence Container. It is important to note that each of these Containers stores detailed information about the Transformations involved in order to get them into their desired form (forming a type of a Transformation History) and the testing and calibration performed in order to ensure that the Transformation performs correctly. These Containers also store the Evidence Containers they are derived from (Figure 13), the generated Proof of Actions and the results from secondary Transformations that are used for verification purposes.

Figure 13: The Transformations of the Keylogger and its Source Code into Evidence Containers (ECs) by the Transformation System



The investigator then uses the Transformation System in order to view the source code and he discovers a number of methods which are of interest. These methods are a capturing method (capturingMethod), a saving method (savingMethod), a transmitting method

(transmittingMethod) and a deletion method (deletionMethod). The capturing method is activated when the Fodacom application begins and contains code to capture login information by capturing the keys pressed during the login process as well as by capturing screenshots during the login process.

When the user presses the login button on the Fodacom application, the capturing method then calls the saving method, the transmitting method and finally the deletion method. The saving method is responsible for the actual storage of the captured login information in text files and in JPEG format for the screenshots once the login button is pressed, which is then followed by a call to the transmitting method. The transmitting method then uses Fodacom's SMTP server in order to send an email containing the captured information, to a DaveCulprit@fodacom.com email address. Unfortunately the investigator was not given access to any SMTP information to support such behaviour, nor was he given access to the employees email accounts. Once the transmitting is complete or should the method fail, a call is then made to the deletion method which is responsible for the deletion of the captured contents from the hard drive. In order to reconstruct and verify this behaviour the investigator once again uses the Transformation System.

The investigator requests that the Transformation System configure a safe, secure virtual environment with a duplicate of the image Container (producing a new Evidence Container containing the virtual state) so that he can reconstruct the actions of the keylogger for verification purposes. The investigator then begins the virtual environment, and takes a screenshot of the active processes, showing KeyLogger.exe at system start up, which is then saved into a new Evidence Container. The investigator then uses the information acquired from the source code and attempts to log into the Fodacom application using a fake username ("investigator") and password ("123456") by pressing the login button. No network connection is provided and, should the email to DaveCulprit@fodacom.com attempt to be sent, it would fail. The investigator uses his knowledge of the source code and then recalls that the captured contents should therefore be deleted, based on the decompiled source code. The investigator then uses the Transformation System in order to attempt to recover any deleted information.

The investigator uses the Transformation System to extract the deleted files, so that the screenshots and associated keylogger text files can be made available to the investigator. The investigator discovers a number of such deleted files dating back to the 1st of April. The investigator however only extracts the screenshots and key logging files that are related to the reconstruction, into separate Evidence Containers named *verification screenshots* and *verification text files*.

The investigator then follows a similar approach to recover and extract the deleted files from the image of the workstation, providing a history of how the key logger has captured Frank Furt's information over the past few weeks, dating back to 1 April 2012. The investigator continues by using the Transformation System to view and extract the text files and screenshots that were created near the time of the incident and discovers a screenshot and text

file created at 10:44 on the 14th May. The investigator then begins forming their arguments to reason about Frank's innocence.

6.6.2.1 Finding Container 1

The first argument the investigator makes (Finding Container 1) is that KeyLogger.exe is indeed a keylogger and compromised Frank Furt's Fodacom authentication information before the incident occurred. The investigator does so by using both the source code as well as the results of the execution of the keylogger in the safe test environment and by providing proof of the keylogger's logging behaviour. These forms of proof include the verification screenshots and text files which are intended to be sent to DaveCulprit@fodacom.com whenever a user attempts to log into Fodacom. The investigator also uses the deleted files contained on the workstation's image in order to show that there has been a history of keylogging. The investigator does this to show that Frank Furt's information has been captured several times before the Maria Madam incident and possibly (as sufficient access to the needed resources is not available) transmitted to a third party.

The investigator argues that the captured information had been transmitted to a third party, to support that the authentication information had been compromised. In order to do so the investigator uses the KeyLogger source code to show that the transmit method was called after the keylogger's capturing process and that the method can be used to transmit the data. The investigator then uses Null Containers to serve as placeholders for the SMTP traffic to support the transmission of such data, and a Null Container to serve as a placeholder for the emails in the DaveCulprit@fodacom.com email account showing proof of the emails. In this manner the investigator is still able to form their argument as well as to capture why the argument cannot be supported at this point in time. The use of these Null Containers therefore help to provide useful guidelines for future analysis and investigators, helping to demonstrate which evidence should be collected and sought after in order to support the Requirement. It may also provide a means to justify to Fodacom why access to the information is needed. In addition, it may provide a means to justify why any access to the email account, particularly by Dave Culprit, should be revoked, as all email information can be easily deleted and hence lost if no sufficient backups and email histories are maintained.

The example helps to demonstrate how investigators can structure their argument using a combination of Evidence Containers as well as Null Containers, and to capture the relationships between them. The Chain of Findings then helps to ensure that all the actions taken by an investigator in their reasoning process are captured and maintained on the investigator's behalf for review purposes and that all information regarding Containers is easily available. In order to emphasize how the Finding Container can aid in capturing an investigator's reasoning process we discuss Requirement Element 1 (e_1) of Requirement 1 (req_1) which states that the KeyLogger.exe was activated at system startup before the day that the incident occurred.

Requirement Element e_1 consists of a $CACQ$ that uses the KeyLogger.exe shortcut Container in order to support req_1 by demonstrating that the shortcut was located in the Startup folder before the day that the incident occurred (d_{CACQ}). The $CACQ$ uses two $ACQs$ to do so, where

the first ACQ (ACQ_1) is used to confirm that the file location attribute of the KeyLogger.exe shortcut, equals that of the Startup folder. The second ACQ (ACQ_2) is then used to confirm that the time of the last modification is equal to the 2nd February 2012 at 08:00 which is before the day of the incident. In this manner the $ACQs$ help to encapsulate how the $CACQ$ achieves its intended goal of supporting the Requirement req_1, and allows for a more detailed recording and reconstruction of the investigator's reasoning process.

Please note each of the $CACQs$ contains an ACQ, that is used to test its admissibility; as a result, we include such an ACQ only where there is no other ACQ which is needed. We have chosen such an approach in order to reduce the size of the example. We have also chosen simple percentage values that we believe help better communicate the example.

Finding Container 1:

$d_{hypothesis}$ = KeyLogger.exe captured Frank Furt's authentication information and compromised it before the incident occurred

$Arg_{supporting}$ = ($H_{ReqC}, nullccwa_{Arg}, d_{conclusion}$) where

- $H_{ReqC} = \{(req_1 \wedge req_2 \wedge req_3 \wedge req_4), 0, d_{ReqC})\}$
- $nullccwa_{Arg} = instanceOf(CCWA_{Null})$
- $d_{conclusion}$ = The evidence demonstrates that Frank Furt's authentication information was compromised by KeyLogger.exe before the incident occurred.

$req_1 = (d_{req}, H_e, rcwa_{req})$

d_{req} = KeyLogger.exe was stored in system Startup before the day that the incident occurred

$H_e = \{e_1, e_2\}$

$rcwa_{req} = instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress **:** 115% / 100% (threshold met)

e	w_e	Details:
e_1:	100%	$CACQ$ = (KeyLogger.exe shortcut, $\{ACQ_1, ACQ_2\}, d_{CACQ}$)
		ACQ_1 = (a = file location, C_s = {a =C:\ProgramData\Microsoft\Windows\StartMenu\Programs\Startup}, d =to confirm that the file location of the KeyLogger shortcut matches that of the Startup folder)
		ACQ_2 = (a = time of last modification, C_s = {a =02/02/2012 08:00, a<14 May 2012}, d =to confirm that the shortcut was last changed on the 2nd of February 2012 at 08:00, which is before the incident)
		d_{CACQ} =The Keylogger.exe shortcut was located in the Startup folder before the incident occurred, allowing the particular program to begin executing as soon as a user has logged into the workstation.
e_2:	15%	$CACQ$ = (Screenshot of KeyLogger.exe in processes at startup, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a =admissible, $C_s = \{a = true\}$, d = confirm that the screenshots are admissible)
		d_{CACQ} = The Keylogger.exe was shown in the processes at system startup during investigator's reconstruction

$req_2 = (d_{req}, H_e, rcwa_{req})$

d_{req} = KeyLogger.exe has the capability to capture a Fodacom employee's authentication information

$H_e = \{e_1, e_2, e_3\}$
$rcwa_{req}$ = $instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress : 100%/100% (threshold met)

e	w_e	Details:
e_1:	50%	$CACQ$ = (KeyLogger Source Code, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a = methods, $C_s = \{\{\text{capturingMethod}\} \subseteq a\}$, d = to confirm that the Source Code contains the specified capturingMethod that contains the code to capture a Fodacom's employee's authentication information)
		d_{CACQ} = in the KeyLogger.exe source code there is a particular method called the Capturing Method that contains code for capturing login information from the Fodacom application
e_2:	25%	$CACQ$ = (Verification Screen Shots, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a = admissible, $C_s = \{a = true\}$, d = confirm that the verification screenshots are admissible)
		d_{CACQ} = The screenshots help to demonstrate the capturing capabilities of KeyLogger and provide proof of the KeyLogger's behaviour
e_3:	25%	$CACQ$ = (Verification text files, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a = admissible, $C_s = \{a = true\}$, d = to confirm that the text files are admissible)
		d_{CACQ} = The text files help to demonstrate the capturing capabilities of KeyLogger and provide proof of the KeyLogger's behaviour

req_3 = $(d_{req}, H_e, rcwa_{req})$
d_{req} = The KeyLogger.exe had captured Frank Furt's authentication information before the incident occurred
$H_e = \{e_1, e_2\}$
$rcwa_{req}$ = $instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress : 100%/100% (threshold met)

e	w_e	Details:
e_1:	50%	$CACQ$ = (The deleted KeyLogger screenshots from image, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a = date, $C_s = \{a \leq 14\ May\ 2012\}$, d = to confirm that the deleted screen shots were taken before the date of the incident namely 14 May 2012)
		d_{CACQ} = The deleted screenshots show Frank Furt's authentication information had been captured before the incident occurred on the 14th May
e_2:	50%	$CACQ$ = (The deleted KeyLogger text files from the image, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a = admissible, $C_s = \{a = true\}$, d = to confirm that the deleted text files are admissible)
		d_{CACQ} = The deleted text files show that Frank Furt's authentication information had been captured

req_4 = $(d_{req}, H_e, rcwa_{req})$
d_{req} = The captured authentication information was transmitted to a third party's email address (DaveCulprit@fodacom.com)
$H_e = \{e_1, e_2, e_3\}$
$rcwa_{req}$ = $instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress : 34%/100% (threshold not met)

e	w_e	Details:	
e_1:	34%	$CACQ$ = (KeyLogger Source Code, $\{ACQ_1, ACQ_2\}, d_{CACQ}$)	
			ACQ_1 = (a = line set, C_s = { [particular line subset]⊆ a}, d = to confirm that the necessary lines are contained within the source code which call the transmitting method, after the storing method, once the user presses the login button. ACQ_2 = (a = methods, C_s = {{transmittingMethod} ⊆ a}, d = to confirm that the source code contains the Transmitting Method which contains code to send the captured authentication information to DaveCulprit@fodacom.com using the Fodacom SMTP server)
			d_{CACQ} =The transmitting method contains code to transmit the captured authentication information to the email address, DaveCulprit@fodacom.com using the SMTP server of Fodacom. This method is then called once the login button is pressed
e_2:	33%	$CACQ$ = (SMTP Log Null Container, $\{ACQ_1\}, d_{CACQ}$)	
			ACQ_1 = (a = lines, C_s= ∅, d = null)
			d_{CACQ} = log entries show that the emails have been sent to the DaveCulprit@fodacom.com email address
e_3:	33%	$CACQ$ = (DaveCulprit@fodacom.com Null Container, $\{ACQ_1\}, d_{CACQ}$)	
			ACQ_1 = (a = email set, C_s= ∅, d = to confirm that the required subset of emails are contained within the emails)
			d_{CACQ} = The emails retrieved from the Dave@gmail.com email address show that the captured information had been received.

6.6.2.2 Finding Container 2

The investigator then formulates an argument to support that Frank Furt's authentication information was compromised and may have been exploited by a fellow employee. In this manner, the investigator can demonstrate that Frank was not involved in the incident and is hence innocent. The investigator formulates such an argument by using Requirements to show that Frank Furt's information was compromised (using Finding Container 1 showing how Finding Containers can build on one another), that Frank Furt was not at the scene of the crime (using the CCTV footage) and that another party logged in. The investigator supports that another party logged in through the use of two Requirements.

The first Requirement states that Dave Culprit had knowledge of Frank Furt's authentication information and the second states that Dave was using Frank's workstation during the time of the incident. The investigator supports the first Requirement through the reuse of the transmitting method ACQ from Finding Container 1, which shows that there was code to send the captured information to a DaveCulprit@fodacom.com email address. The investigator then reuses a $CACQ$ as a Requirement element to represent the SMTP Null Container which, should the actual evidence be made available, can be used to show that the email was indeed sent to the DaveCulprit@fodacom.com email address. The investigator continues by using a DaveCulprit@fodacom.com Account Null Container to show that the owner of the account

must be shown to be Dave Culprit. In this manner the investigator captures his reasoning process and states that, in order to support this Requirement (and hence the argument), not only must proof be shown that the captured information was sent to DaveCulprit@fodacom.com email address, but that the email address must also be shown to belong to Dave Culprit.

In order to support the Requirement that Dave was using Frank's workstation during the time of the incident, the investigator uses CCTV footage, Dave's fingerprints found on the workstation, user log records and the captured information by KeyLogger.exe near the time of the incident (which was deleted). The CCTV footage shows Dave working on Frank's workstation during the time of the incident, and the fingerprints help to support that Dave did interact with the workstation. The user log records show that Frank logged in during this period, while the CCTV footage shows that Frank was not present. The user log records also support that the login came from Frank's workstation through the recorded static IP address and MAC address. This is further supported by the recovered screenshots and text files that were created and deleted by the KeyLogger during the alleged period of the incident, providing greater certainty for the Requirement. The investigator then uses all of these Requirements to formulate their argument and to support Frank Furt's innocence. However, the certainty that the investigator can place into their argument is strongly dependant on the reliability of the evidence that the investigator uses.

Finding Container 2:

$d_{hypothesis}$ = Frank Furt's authentication information was compromised and may have been exploited by a fellow employee to perform the SIM swap

$Arg_{supporting}$ = (H_{ReqC}, $nullccwa_{Arg}$, $d_{conclusion}$) where

- $H_{ReqC} = \{(req_1 \wedge req_2 \wedge req_3 \wedge req_4), 0, d_{ReqC})\}$
- $nullccwa_{Arg} = instanceOf(CCWA_{Null})$
- $d_{conclusion}$ = The evidence demonstrates that Frank Furt's authentication information was compromised and may have been exploited by a fellow employee to perform the SIM swap

req_1 = $(d_{req}, H_e, rcwa_{req})$

d_{req} = Frank Furt's authentication information was compromised

$H_e = \{e_1\}$

$rcwa_{req}$ = $instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress **:** 100%/100% (threshold met)

e	w_e	Details:	
e_1:	100%	$CACQ$ = (Key Logger Argument, $\{ACQ_1\}, d_{CACQ}$)	
		ACQ_1 = (a =supported, $C_s = \{a = true\}$, d = to confirm the argument is sufficiently supported)	
		d_{CACQ} = the Finding Container helps to demonstrate that Frank Furt's authentication information was compromised.	

req_2 = $(d_{req}, H_e, rcwa_{req})$

d_{req} = Frank Furt was not present at the time of the incident

$H_e = \{e_1\}$

$rcwa_{req}$ = $instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress **:** 100%/100% (threshold met)

e	w_e	Details:
e_1:	100%	$CACQ$ = (CCTV footage, $\{ACQ_1\}$, d_{CACQ})
		ACQ_1 = (a =time period, C_s = {[14/05/2012 10:00 – 14/05/2012 11:00] $\subseteq a$, d =to confirm that the time from 14/05/2012 10:00 to — 14/05/2012 11:00 is included in the CCTV footage
		d_{CACQ} = on 14 May 2012 at 10:00 Frank is shown leaving the scene and does not return until 11:00.

$req_3 = (d_{req}, H_e, rcwa_{req})$

d_{req} = Dave had knowledge of Frank Furt's user authentication
H_e ={e_1, e_2, e_3}
$rcwa_{req}$ = $instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress **:** 40%/100% (threshold not met)

e	w_e	Details:
e_1:	40%	$CACQ$ = (KeyLogger.exe Source Code, $\{ACQ_1\}$, d_{CACQ})
		*ACQ_1 = (a = methods, C_s = {{transmittingMethod} $\subseteq a$}, d = to confirm that the source code contains the Transmitting Method, which contains code to send the captured authentication information to DaveCulprit@fodacom.com using the Fodacom SMTP server) **Please note * is used to represent reuse**
		d_{CACQ} = Source code, method showing the transmitting of captured information via email to an employee email address Dave@gmail.com
e_2:	30%	*$CACQ$ = (SMTP Log Null Container, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a = lines, C_s= $\emptyset$, d = null)
		d_{CACQ} = log entries show that the emails have been sent to the DaveCulprit@fodacom.com email address
e_3:	30%	$CACQ$ = (DaveCulprit@fodacom.com Null Container, $\{ACQ_1\}, d_{CACQ}$)
		ACQ_1 = (a = owner, C_s = {a = Dave Culprit}, d = to confirm that the email account belongs to a Dave Culprit)
		d_{CACQ} = The email address DaveCulprit@fodacom.com belongs to Dave Culprit

$req_4 = (d_{req}, H_e, rcwa_{req})$

d_{req} = Dave Culprit interacted with laptop during period of incident
H_e ={e_1, e_2, e_3, e_4}
$rcwa_{req}$ = $instanceOf(RCWA_{PB})$ such that the desired certainty threshold is 100%

Certainty Progress **:** 140%/100% (threshold met)

e	w_e	Details:
e_1:	20%	$CACQ$ = (Dave Culprit's fingerprints on workstation, $\{ACQ_1\}$, d_{CACQ})
		ACQ_1 = (a =source, C_s = {a = Dave Culprit}, d = to confirm that the fingerprints belong to Dave Culprit)

						d_{CACQ} = the fingerprints found on Frank's workstation shows that Dave interacted with the workstation at some point in time
			e_2:	40%	$CACQ$: (CCTV footage, $\{ACQ_1\}$, d_{CACQ})	
						ACQ_1 = (a =time period, C_s = {[14/05/2012 10:44 – 14/05/2012 10:46] ⊆ a, d = to confirm that the CCTV footage does contain the necessary footage between the specified time period)
						d_{CACQ} = the video footage shows Dave Culprit using Frank's workstation between 10:44 and 10:46 on the 14th of May 2012, which is during the time of the incident.
			e_3:	40%	$CACQ$ = (Fodacom's User Login line x, $\{ACQ_1\}$, d_{CACQ})	
						ACQ_1 = (a = user, C_s = {a = Frank Furt}, d = to confirm the user logged in is Frank Furt) ACQ_2 = (a =time, C_s = {a =14/05/2012 10:44}, d =to confirm that the login occurred at the specified time)
						d_{CACQ} = The user log line shows that Frank Furt logged in at a 10:44
			e_4:	40%	$CACQ$ = (14 May 2012 10:44 deleted authentication information, $\{ACQ_1\}$, d_{CACQ})	
						ACQ_1 = (a = time created, C_s = {a =14/05/2012 10:44}, d = to confirm that the authenticated information was captured just before the incident
						d_{CACQ} = The deleted, captured authentication information shows that a login attempt was made from Frank's workstation a minute before the SIM swap was done.

While the example does capture inculpatory evidence and how such evidence can be used to support a Finding Container, an investigator must take into account exculpatory evidence and evidence of tampering as well. By doing so an investigator can ensure that their arguments are more objective and overall more sound. A possible refuting argument could be used to show that Frank's authentication information was used and was not compromised — showing that Frank Furt may have disclosed his password (willingly) and is hence accountable for this disclosure. Appropriate evidence can then be linked to the Requirements of the argument and used to show the argument should therefore be denied. An investigator must also take into account evidence of tampering. For example, Frank may have set the workstation's date back, installed the keylogger, created the fake DaveCulprit@fodacom.com email address, and executed the KeyLogger.exe a number of times to generate the planted information to frame Dave. Frank may also have tampered with the CCTV footage, perhaps overwriting its contents with old footage, making it seem as if Dave was using his workstation during the alleged period of the incident. It is therefore important that the investigator take into account Frank's experience and skills with regards to computing systems, and that the investigator confirms the reliability of the evidence sources that they use.

6.6.3 Example Discussion

Our intention for this fictional scenario was to demonstrate how an investigator could possibly use the Chain of Findings to formulate an argument. Investigators are by no means

limited to such an approach, as the model provides the necessary flexibility for investigators to structure their arguments in the manner that best suits their reasoning process. Investigators may therefore choose different weighting schemes and choose more efficient means for forming their arguments. Investigators can decompose their arguments into smaller arguments and combine these smaller arguments or simply use one large argument. Investigators can also change the Requirements and combine them as they see fit and add new Requirement elements or remove them. In this manner, the model helps provide investigators with the flexibility to formulate their arguments as they wish, while still ensuring that these arguments adhere to the well defined structure specified in Chapter 5. Overall, flexibility in an investigator's reasoning process is beneficial and is demonstrated in the example, but the example also helps to emphasize how Containers are used within the model.

The example helps to show how various Evidence Containers can be used uniformly as building blocks to form an investigator's arguments and how detailed information about how these building blocks came to be is captured, along with the investigator's reasoning process. The information includes the Evidence Containers that the building blocks are derived from, the Transformations involved and their associated testing (test history) which show the reliability of the Transformations that produced these Containers. In addition, the investigator is given the capability to use description elements along with these building blocks so that they can not only capture their reasoning process, but also the rationale behind it. The detailed audit trail captured by the model hence allows for a more thorough reconstruction process with regards to an investigator's building blocks and their reasoning process. The detailed audit trail also uses Proof of Actions so that the audit trail provides non-repudiation and captures the time that the various events took place. Doing so allows for a timeline of events to be created, allowing for a more through reconstruction. The Chain of Findings and the Finding Containers also aid Fodacom in terms of Digital Forensic Readiness.

The Finding Containers, whether used as a template for organisations or simply as a means for an organisation to draw up potential scenarios, helps ensure that organisations can better achieve Digital Forensic Readiness. Using the example, one can see how Fodacom is shown the potential sources of evidence which may be relevant in these scenarios, such as the SMTP servers, emails, network traffic and user audit information. By providing Fodacom with information about these sources before an incident occurs, Fodacom can be better prepared to ensure that these sources are protected, properly preserved, available when they are needed and reliable. In this manner investigations can occur faster and more efficiently.

The fictional scenario also shows the importance of strong authentication, backup generators and anti-virus software that is always up to date. The fictional scenario and the Finding Containers emphasize the need for stronger forms of authentication, particularly for important actions such as SIM swaps. Approaches such as smart cards and/or biometrics would have helped to prevent Frank's authentication information from being fabricated so easily and would provide a stronger form of non-repudiation. Organisations are shown the importance of backup generators through the various usage of the CCTV footage which, should the power have gone out, may not have been available. The anti-virus software is important as it

could have detected the keylogger, or more likely the keylogger's malicious behaviour. The anti-virus software could have then taken the appropriate actions, such as moving the .exe to quarantine or simply deleting it, and notifying the appropriately registered parties about the keylogger's presence.

6.7 Chapter Summary

In this chapter we discussed our model and motivated how it can aid investigators and investigations. We discussed the detailed audit trail captured by the Chain of Findings throughout the reasoning process and how it allows for a more thorough reconstruction. We also explored the benefit of the feedback that the model provides with regards to an investigator's progress. In addition, we elaborated on the flexibility of the model and its adaptive capabilities to cater for future needs and Requirements within the Digital Forensics field. We then discussed a further benefit of the model, namely the model's capability to aid in achieving Digital Forensic Readiness. In section 6.6 we then provided an example of the model in a fictional scenario.

The fictional scenario allowed us to elaborate on how the Chain of Findings can aid an investigator in an investigation, and provide an example of its usage. The example allowed us to discuss how an investigator could capture their reasoning process, the steps they followed and the building blocks from which their reasoning is based. The example allowed us to discuss the various uses and benefits of the smaller components in Finding Containers and the model in general, such as the description elements and the $ACQs$. The fictional scenario also allowed us to demonstrate how investigators can build on their existing arguments and how they can capture the certainty that they place in these arguments. In addition, we illustrated how an investigator can use Null Containers to serve as placeholders, providing guidelines for which information needs to be searched for and acquired in an investigation. In the next chapter we discuss the prototype for the Chain of Findings, which serves as a proof of concept for the model.

Chapter 7

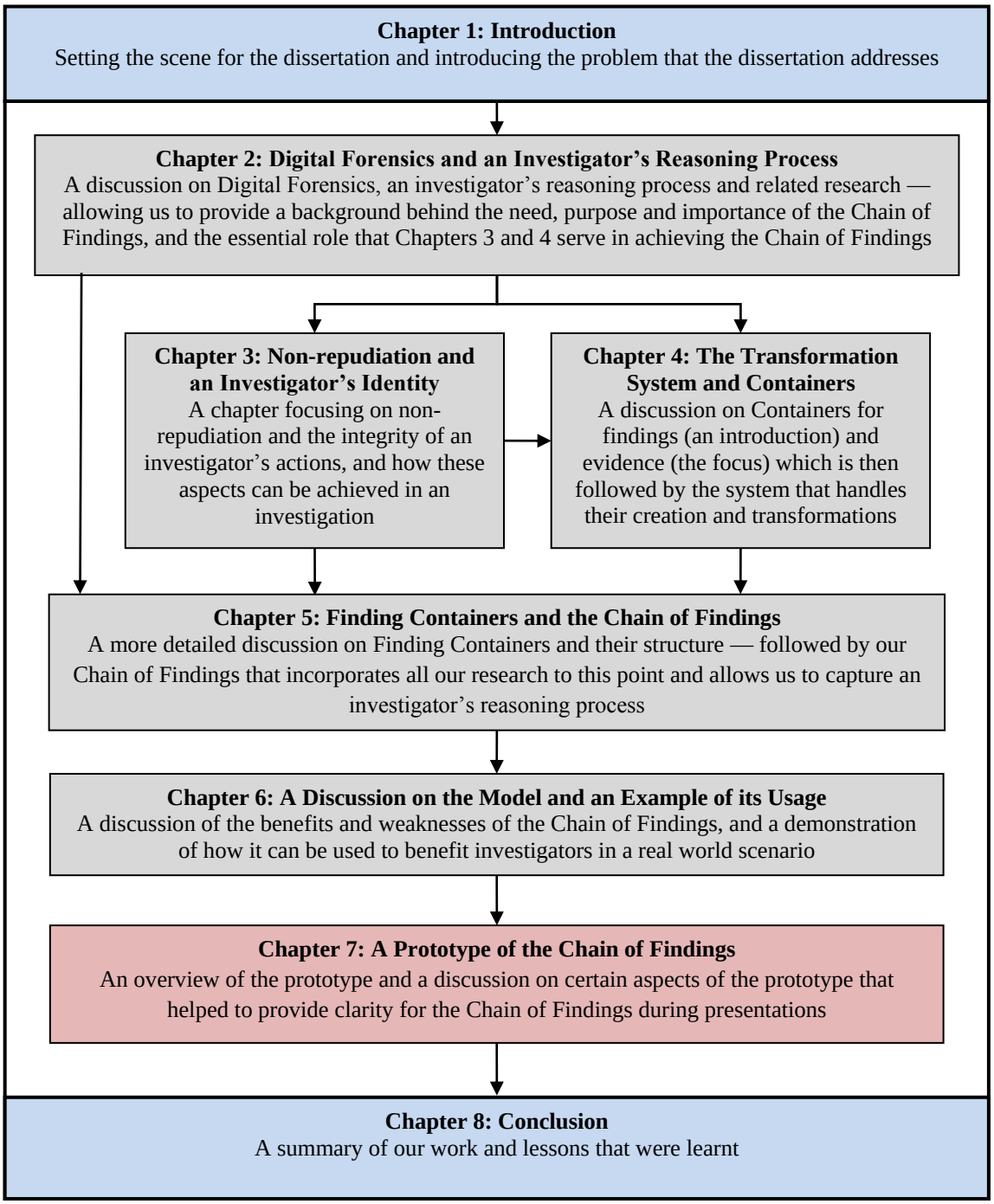
A Prototype for the Chain of Findings

Objectives

- To provide a summary of the prototype for the Chain of Findings and its purpose in our research
- To elaborate on certain aspects of the prototype that helped to provide clarity for the Chain of Findings during presentations and to briefly discuss how these aspects were implemented

A life spent making mistakes is not only more honourable, but more useful that a life spent doing nothing
George Bernard Shaw

A Map of our Current Location in the Dissertation



Chapter 7: A Prototype of the Chain of Findings

7.1 Introduction

In this chapter we explore the prototype, which was created to serve as a proof of concept for the Chain of Findings and a basis for future implementations. Throughout our research the prototype served another important role, namely as a demonstrational tool for the Chain of Findings. The prototype helped to demonstrate to various parties in our research group how the various components fit together and it helped them to better understand the Chain of Findings. The prototype also helped to communicate the various uses and capabilities of the Chain of Findings.

Figure 14: The Login

The prototype is coded using Java (version 1.6.0) through the use of the Netbeans IDE (version 6.9.1) and it uses a MySQL database to store data. The MySQL service is provided through the use of Wampserver (version 2.0) running MySQL (version 5.1.36). Once the prototype is started, an investigator is presented with a login screen as shown in Figure 14. The prototype incorporates nine fictional investigators having one of four fictional ranks. Once the investigator successfully logs in they are then presented with a Welcome Screen, which is demonstrated in Figure 15. On the Welcome Screen the investigator can choose to view information about investigators working on the case, enter evidence and view and transform the Containers related to the case. To simplify the prototype we restricted the number of cases to only one, and the number of actively logged investigators to one. We also restricted the $RCWA$s and $CCWAs$ to only Percentage-Based and Null-Based, and $CACQs$ to only apply to Evidence Containers. Furthermore, we restricted the attribute types to String, Integer and Date. In order to help accelerate the creation of the prototype, the prototype reuses the GUI theme and the user data created for a previous research project coded by the author. The user data is used in the prototype to serve as fictional investigators in the prototype. Furthermore, the prototype only focuses on capturing provenance information about digital events regarding Transformations, showing that provenance information can be captured for Containers. The prototype is however composed of 36 packages and 208 class files, and we attempted to incorporate as much of the functionality as possible, while also ensuring a reusable design and that the interface provides all the necessary information in an effective manner.

Figure 15: The Welcome Screen

In the remainder of this chapter we provide an overview of some of the aspects of the prototype, how they work and how we chose to implement them. The aspects we chose to discuss are based upon the queries which were most prominent in different presentations to our research group. By elaborating on these aspects we hope to provide clarity and answers for readers about similar queries and we hope to provide a basis on which future implementations of the Chain of Findings can be built upon. In each of the sections in this chapter, we first provide a summary of what the particular aspect is, what the query was and then we elaborate on the answer and how it was implemented.

In section 7.2 we elaborate on $ACQs$ and their criteria set, allowing us to explain and demonstrate what criteria is and how criteria can be specified. We also discuss how the prototype allows investigators to specify attributes that a Container does not have, and we provide an overview of their implementation in the prototype. We then continue in section 7.3 with our discussion on the Transformations we incorporated in the prototype and the Proof of Actions that they generate. Section 7.3 allows us to provide concrete examples of potential Transformations and to demonstrate the benefits of a Proof of Action History. In the next section we elaborate on how an investigator can combine Requirements using logical connectives and brackets (to formulate $combination_{req}s$), demonstrating how $combination_{req}s$ can be achieved in future implementations. Finally we conclude the chapter in section 7.5.

7.2 $ACQs$, the Criteria Set and Null Attributes

An Attribute Checking Query or ACQ (Section 5.3.1) is used by an investigator to capture an attribute of a Container, and the particular criteria that the attribute's value should comply with to serve a particular role in the investigator's reasoning. $ACQs$ in the prototype can apply to any attributes of an Evidence Container as well as attributes that a Container may not necessarily have, referred to as Null attributes in the prototype. The types of attributes incorporated into the prototype are String, Integer, Boolean, Date and Null. These attributes each have a name, a description and a particular value. The name is used to represent what the particular attribute is, the description describes what the attribute is for and the value represents the attribute's value. The Null attributes are used to allow an investigator to specify the attribute that a Container does not have, but that is needed and must meet certain criteria in order for the Container to serve its role in the investigator's reasoning process. Each of these attribute types is implemented as a subclass of a super class called Attribute, ensuring that they can all be used uniformly, using polymorphism.

The criteria for the attribute are specified by the investigator using a text field and a combo box that allows the investigator to specify whether the attribute should be $<, \leq, =, \geq$ or $>$ than the specified value. Each of the attributes then has a isCriteriaValueValidForAttribute() method, inherited from the super class Attribute, which returns true should the value be applicable to the particular attribute. Should the specified value not be applicable to the attribute, a particular exception that contains details of why the value is not applicable is thrown. The exception is then reported to the investigator, allowing them to correct the

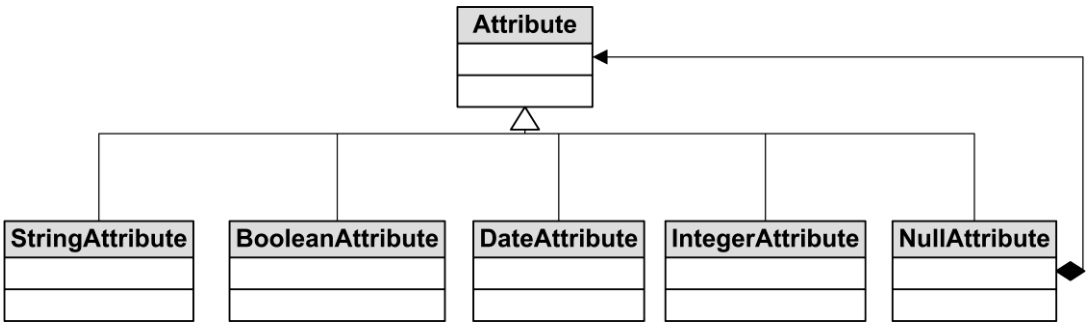
entered value. In this manner the isCriteriaValueValidForAttribute() is used to confirm that the specified value falls within the domain of the attribute. Should there exist more specialised attribute types, requiring more distinct domains, they can then simply inherit from the appropriate attribute and override the isCriteriaValueValidForAttribute() method. The investigator can continue entering such criteria until they have completed the set of criteria that the attribute must comply with. It is important to note that in the prototype notifies the investigator should he enter duplicate criteria. The prototype helped to clarify what was meant by criteria, and provided an example of how it could be specified to the research group. Now that we have discussed attributes in greater detail, we discuss the Null attribute in more detail.

Figure 16: The ACQ JFrame and Entering of Criteria

Null attributes are designed to resemble the attributes that a Container does not have, so while they do not necessarily have a value, we wanted to ensure that the criteria the investigator specifies is still appropriate for the intended attribute. One approach would be to create a Null attribute for each particular attribute used in the prototype, for instance for the File Hash attribute we could have a Null File Hash Attribute. Unfortunately, such an approach results in significant duplication and the real world Chain of Findings will have to cater for a large number of attributes. In order to use a more efficient approach, we use the Decorator pattern (discussed in [26] and [36]), which allows us to use the Null attribute to wrap the actual attribute type. The Null attribute then uses the wrapped attribute to provide the necessary concrete behaviour. For example, when isCriteriaValueValidForAttribute() is called on a Null

attribute, it then executes the isCriteriaValueValidForAttribute() method on the wrapped attribute, and returns its value — therefore mimicking the behaviour of the wrapped attribute. We also used a similar approach to append "<null>" to the beginning of the attribute's name and description, to emphasize to the investigator that the attribute they are using is not part of the Container. Should any criteria be tested against the Null attribute, it simply returns false — ensuring that criteria against a Null attribute can never be met and that all such criteria tests will fail. The Decorator pattern therefore helps to create the illusion that there are indeed Null versions of each attribute, while providing a more efficient approach for doing so. The UML for the attributes and the decorator pattern can be seen in Figure 17.

Figure 17: The Decorator Design Pattern



7.3 Transformations in the Prototype

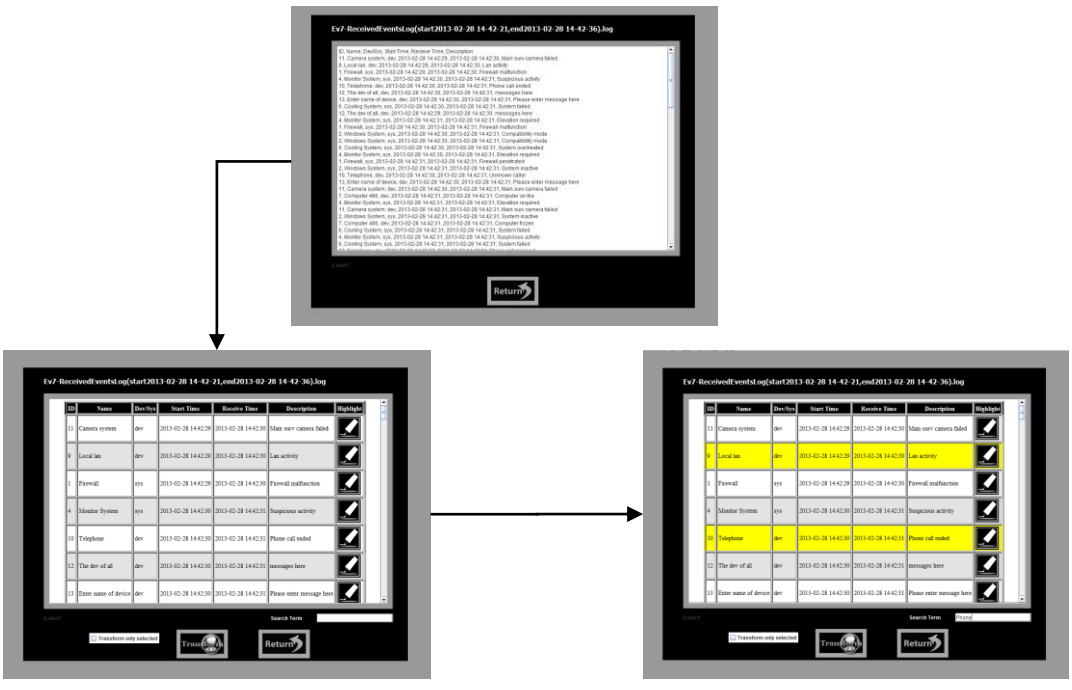
In the prototype, inheritance plays an important role throughout the implementation process. Inheritance helps to ensure that behaviour is reused and that subclasses can build on the behaviour of the parent classes in order to provide more specialised behaviour. It also allows us to use the Template design pattern to specify how certain actions should be performed, the steps required and the sequence they should be performed in, while still providing the flexibility for subclasses to specify how these steps should be performed. One particular use for such a template was during testing and Transformations, allowing the super classes to provide a skeleton of steps for how testing and Transformations should be performed. Subclasses of these classes, namely concrete Transformations, are then given the flexibility to specify how these steps should be performed while still ensuring that they adhere to the sequence in which the steps should be performed. In the presentations, what was meant by Transformations was not clear to certain audience members and the prototype helped to provide concrete examples of what a Transformation (Chapter 4) is, and how they can be used by investigators. We now discuss the Transformations used in the prototype in more detail.

The Transformations we incorporated into the prototype apply to text files (.txt), generated comma-delimited CSV log files and picture formats (.jpg, .png and .bmp). The generated comma-delimited CSV files are used to represent fictional log files and are incorporated to allow us to emphasize the importance of Transformations. The Transformations that can be used with these particular file formats include, a forensic image and transform

Transformation, a standard view Transformation and a rich view Transformation. The forensic image and transform Transformation is used to allow an investigator to enter particular files that are of interest and to create forensics images verified through the use of an MD5s. These forensic images are then placed into Evidence Containers, configured with the appropriate attributes and a unique ID. The standard view Transformation is used to allow an investigator to view the text files and the generated log files in a standard text format. The standard view Transformation can also be used for picture files, allowing an investigator to simply view the associated .jpg, .png or .bmp image. The standard view Transformation helped to emphasize that even viewing a particular file requires a Transformation to transform its contained bytes to a viewable format, due to the latent nature of evidence.

The rich view Transformation is then used to read the contents of a particular Evidence Container and transform it to a richer format, providing more advanced capabilities. For text files, each line is displayed in a table with a line number in the first column followed by the line's contents in the second column. The investigator is then able to search for relevant keywords in the text file. Each instance of these keywords that is found in the text is bolded and underlined (for ease of reference), and the line is highlighted. Investigators can then choose to transform this evidence into a new Evidence Container storing only the highlighted lines, or storing all the lines and maintaining information on the highlighted lines, while preserving the original. Similar viewing and transformation capabilities are provided for the generated CSV files and can be seen in Figure 18. The rich view Transformation demonstrates the benefits that Transformations can offer to investigators for instance how it can aid them in highlighting, searching and retrieving certain, relevant data.

Figure 18: The View Transformations of a Log File



For images (.jpg, .png and .bmp), the rich view Transformation allows investigators to tag particular areas of interest in the image using a bounding box and to associate a comment

with the bounding box. Investigators are then given the capability to view, update, and delete these comments should they wish. When the investigator is satisfied with the added comments the investigator can then produce a new Evidence Container containing their comments and specify the title for the new Evidence Container. The titles allow the investigator to more easily locate Evidence Containers. The rich view Transformation for pictures can be seen in Figure 19 and demonstrates the variety of Transformations that can benefit an investigator.

Figure 19: Viewing Transformations for Image Files

Figure 20: Viewing the Investigator Accountable

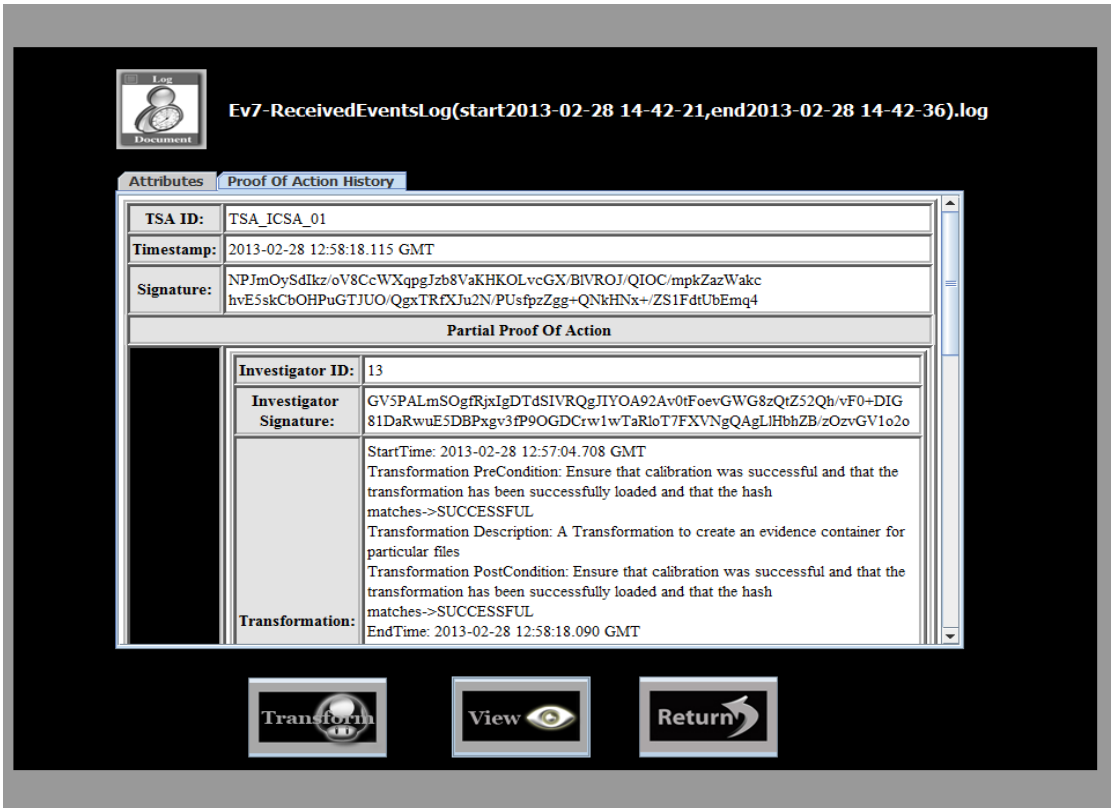
In the prototype we also demonstrated the benefits of capturing the investigator who is responsible for performing these Transformations. This was shown in the presentations by

navigating to a particular Evidence Container of interest and clicking on the investigator responsible for creation attribute, which then opened up a new JFrame displaying the investigator. Should more in depth information regarding the investigator responsible be required, one can click on the highlighted thumbnail (Figure 20) to view this information. This information includes the investigator's name, surname and contact details, as well as their public key for the case — for verification purposes. We used this to demonstrate some of the benefits of a detailed audit trail, which resulted in further queries regarding the Proof of Action History for these Evidence Containers. In addition, when one hovers over a particular attribute, semantic information regarding the attribute is given, demonstrating the benefits of having an attribute's meaning and purpose explicitly defined.

7.3.1 The Proof of Action History

The Proof of Action History for any particular Container maintains all Proof of Actions (Section 3.5) regarding the particular Container. In order to create a Proof of Action, an initial Partial Proof of Action is created, consisting of an investigator's ID and the action the investigator performed. This Partial Proof of Action is signed using the investigator's private key. The Partial Proof of Action is then timestamped by a Time Stamping Authority (TSA) and signed by the TSA to produce a Proof of Action.

Figure 21: The Proof of Action History



In order to demonstrate and clarify the purpose of such a history, we made each particular Transformation produce a Proof of Action that contains detailed information about its fictional test history and its particular Transformation process (Figure 21). The details of the

Transformation included precondition actions (such as ensuring that the hash of the Evidence Container is correct) and postcondition actions (such as verifying that the hash of the Evidence Container did not change). The Transformation information also included information such as when the action began, when it ended, and what particular actions the investigator did while performing the Transformation, such as adding comments and searching for key words. The generated Proof of Action is then timestamped by a TSA that uses the time of the system on which the prototype is executed and a standard format for recording the time. The generated Proof of Action is then added to the Proof of Action History of the Evidence Container and is viewable in the prototype.

The creation of these Proof of Actions was demonstrated a number of times using a number of Transformations, such as a simple view Transformation that recorded when the investigator started the viewing process and when they ended it. The demonstration emphasized the benefits of a Proof of Action History and how it helps to capture and maintain detailed provenance information for the Evidence Container. The demonstration therefore helped to clarify how such a history can be used to substantiate the integrity and reliability of evidence throughout the investigation. The demonstration also helped to show the benefits of a uniform time format and how the Proof of Action History can be used to create a timeline of when actions were performed, benefiting the detailed audit trail and allowing for a more thorough reconstruction process.

7.4 The Combinations of Requirements and their Parsing

In our discussion of the Chain of Findings we discussed the Requirement Combination, a tuple of the form $(combination_{req}, w_{ReqC}, d_{ReqC})$. The Requirement Combination consists of a combination of Requirements — that are combined using logical connectives ($combination_{req}$) and provide confirmation for an Argument — its weight (w_{ReqC}) and its description (d_{ReqC}). One particular aspect that was queried during the presentations of the prototype, was the use of $combination_{req}$ and how an investigator goes about specifying such a combination.

In order to allow an investigator to create these $combination_{req}$s, an investigator must be able to freely combine Requirements using logical connectives and brackets. To achieve this in the prototype, each particular Requirement Combination has a particular ordered list which investigators can add and remove Requirements to and from. This set represents the particular Requirements that an investigator would like to use to formulate their $combination_{req}$. Each Requirement that is added to this list is given an ID, based on its position in the list. The ID is unique to the Requirement Combination to aid the investigator in identifying and specifying particular Requirements and in forming their particular $combination_{req}$. Whenever an investigator clicks on a particular Requirement, the Requirement's ID is then added to the $combination_{req}$ text field at the current caret position. An investigator can then use the various logical connective buttons and mouse clicks on the Requirements of interest to

formulate their $combination_{req}$ in the text field. Investigators can also enter the $combination_{req}$ in the text field using the keyboard, or a combination of both approaches.

In order to evaluate the newly specified $combination_{req}$ and ensure that it parses correctly we used an expression language called MVEL (available from http://mvel.codehaus.org/Home). MVEL was chosen as we were familiar with the language and had used it previously, however any expression language could be used. MVEL takes the particular Requirements and the results of their evaluations to determine whether the specified combination of Requirements holds. We used MVEL to perform the parsing and the evaluation of the combination of Requirements through the use of Java's HashMap<String, Boolean>. The String values represent the Requirement IDs and the Boolean values represent the results of the each of the Requirements' evaluations. The combination is then parsed and evaluated by executing MVEL.evaluate($combination_{req}$, HashMap<String, Boolean>). Should MVEL detect any errors while parsing the $combination_{req}$, it will then throw an Exception, which is displayed to the investigator, allowing them to correct the entered $combination_{req}$ accordingly. The investigator is then only allowed to commit the entered $combination_{req}$ if there are no parsing errors.

Figure 22: The Requirement Combination JFrame

7.5 Chapter Summary

Our intention for this chapter was to use our feedback from the prototype to demonstrate what certain components are, how these components fit together and how they can be implemented. We hope that future developers will build upon the prototype and use our discussions as a background so that we can come closer to achieving the Chain of Findings in the near future.

Chapter 8

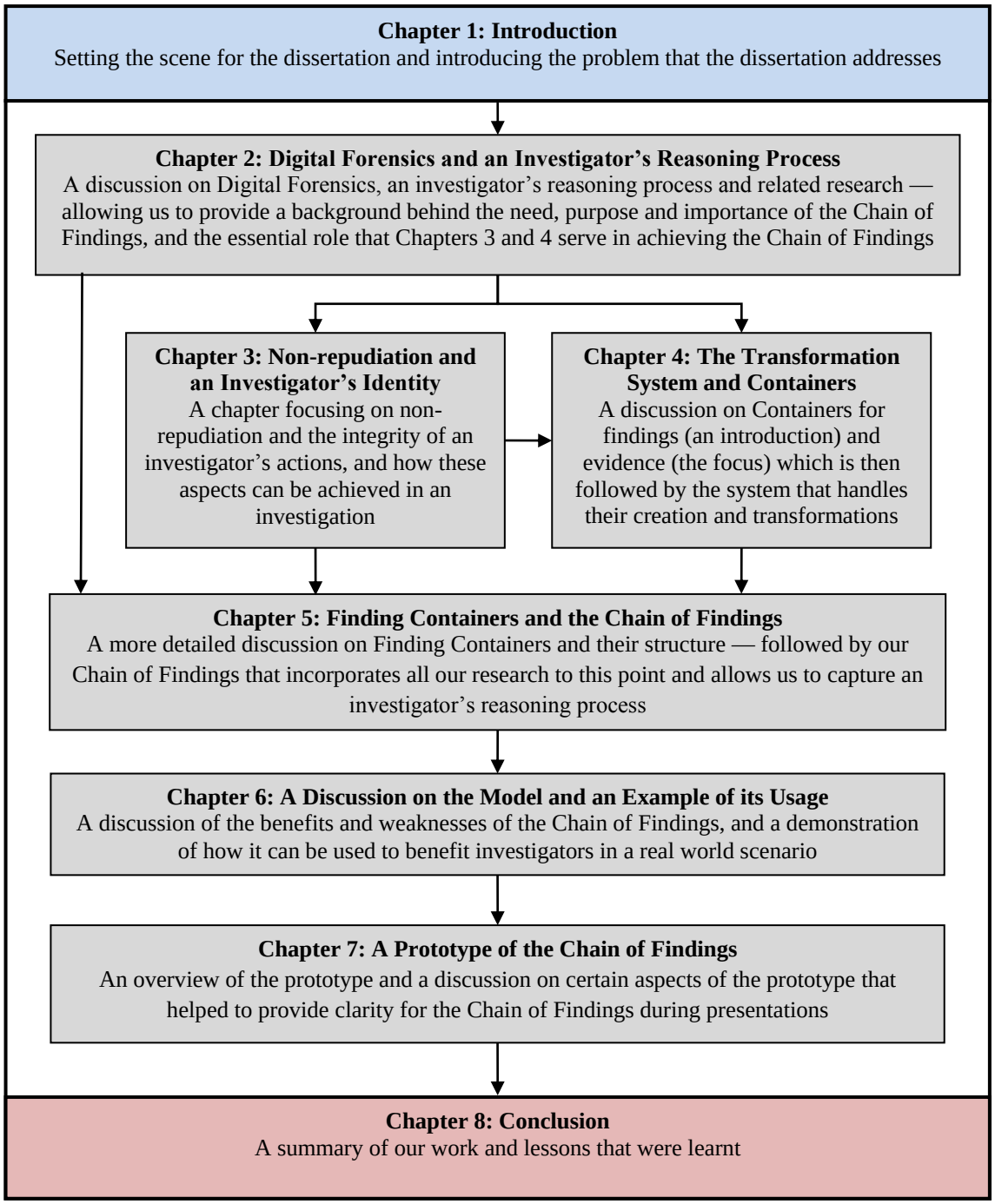
Conclusion

Objectives

- To provide a summary of the model, its creation process and what was learnt from this dissertation

It's better to light a candle than to curse the darkness
Eleanor Roosevelt

A Map of our Current Location in the Dissertation



Chapter 8: Conclusion

In this dissertation our main focus is to determine whether it is possible to create a Chain of Findings. The main intention for the Chain of Findings is to provide a formal means to capture and express an investigator's reasoning process and to provide feedback on the impacts of the dynamic aspects of an investigation. In Chapter 1 we set a number of requirements that we believe are essential for the Chain of Findings. We explore these requirements and discuss whether they were achieved. Doing so allows us to measure whether we achieved our goal and where more research is required for our model. We begin by restating the requirements we set for the Chain of Findings in Chapter 1.

An investigator must be able to capture their conclusions, the arguments behind their conclusions and the rationale behind each of the steps of their reasoning process. The Chain of Findings must provide flexibility in the manner in which investigators can formulate their arguments, factor in certainty and achieve their desired findings. The Chain of Findings must capture all details of an investigator's reasoning process, including the combination of building blocks that an investigator uses for their arguments. These building blocks may be evidence — digital or physical — and existing findings. Investigators must be able to use these building blocks uniformly and capture the attributes and criteria that allow these building blocks to serve/not serve their roles in the investigator's reasoning. The Chain of Findings should incorporate a means for investigators to reliably transform their building blocks, as is needed, and to record a detailed audit trail of the Transformation and its reliability. The Chain of Findings should also allow for the capturing of provenance information regarding not only the Transformations performed but also physical events, and it should ensure that non-repudiation is a core aspect of all recorded events. The Chain of Findings must also allow investigators to incorporate multiple confirmations for their reasoning, and it must allow investigators to capture alternate explanations and show why they cannot hold. Finally, the Chain of Findings should be designed with flexibility in mind to allow it to grow and adapt with the advancements in digital devices and media. Before we discuss our model for the Chain of Finding, let us begin by first discussing one of the most vital aspects of the model, namely the Finding Container.

Finding Containers allow investigators to capture and formulate the reasoning behind their conclusions using a combination of goal-driven and data-driven approaches. In addition, they allow investigators to capture the rationale and certainty behind their reasoning. Each Finding Container allows an investigator to capture their hypothesis as well as their supporting and refuting argument for the particular hypothesis. Each of these arguments allows an investigator to capture multiple confirmations for why the argument should hold/not hold and they capture how the investigator reasoned about their hypothesis using inculpatory evidence, exculpatory evidence and evidence of tampering. In this manner an investigator is also able to specify and capture alternate explanations and show why they do not hold. Should both the supporting argument and refuting argument hold, the Finding Container incorporates a Reasoning Algorithm. The Reasoning Algorithm allows an investigator to capture which

argument should get preference (such as the one with the highest certainty) and the rationale behind their encoded reasoning.

Investigators can use evidence and existing findings uniformly as building blocks for their reasoning. Investigators can capture not only the attributes and criteria that allow/do not allow evidence and existing findings to serve their role, but also the certainty they contribute and the rationale behind them. For evidence, these attributes and criteria include whether the evidence must be admissible, and for findings the criteria includes whether the finding must be supported, refuted or neither supported nor refuted in order to serve its role in the investigator's reasoning. Should these attributes' values change, then these criteria checks will no longer return the same result. Finding Containers use these checks along with the certainty they contribute to provide feedback on the impacts that dynamic aspects have on their finding and, overall on an investigator's reasoning.

The Finding Container allows investigators to incorporate certainty into different components of their reasoning, not just their building blocks — providing investigators with greater flexibility with how they factor certainty into their reasoning. In addition, investigators are given the freedom to choose how best to reflect and reason about certainty in these components through Certainty Weighting Algorithms. One particular Certainty Weighting Algorithm, namely the Null Certainty Weighting Algorithm, provides investigators with the capability to ignore certainty in their reasoning and/or in selected components, providing greater flexibility for the investigator. Further research is however required to determine what the best representation for certainty is in different scenarios and how an investigator should best allocate certainty weights to components based on the certainty they contribute, particularly for the investigator's building blocks. These building blocks, evidence and finding-based, are captured in Containers created and transformed by the Transformation System.

The Transformation System is used for the creation, viewing and transforming of Containers for evidence and findings. Evidence Containers are the main focus of Chapter 4 and are particular Containers that allow all evidence, digital and physical to be treated uniformly by investigators. In order to allow the Transformation System to cater for new digital devices and media, the Transformation System incorporates registration capabilities. These registration capabilities provide a means for new Transformations to be easily added to the Transformation System. Finding Containers are similar in structure to Evidence Containers and allow investigators to capture and represent their findings. The Transformation System is responsible for configuring these Containers' attributes, using an Attribute Class Repository. The use of such an Attribute Class Repository helps to ensure that attributes' syntax and semantics are explicitly defined and are used consistently through all Containers that require such an attribute — benefitting an investigator's captured reasoning. In Chapter 6 we discussed how the use of an Attribute Class Repository might also be a drawback for the Chain of Findings, particularly due to the important role it plays with regards to the configuring of these Containers. Should any changes be required to the attribute classes specified in such a repository, it will impact all Containers that are configured with attribute instances (referred to as attributes) of these classes. These changes may then also impact the

attributes, criteria and rationale for why and how these Containers serve their role in an investigator's reasoning. Because of these reasons, the Attribute Class Repository and its various attribute classes must be well accepted (and understood) in the Digital Forensic community. Whenever the Transformation System performs a Transformation (on behalf of an investigator) on a particular Container, it records the Proof of Action into the Container's Proof of Action History.

In Chapter 3 we explored the creation of a Proof of Action that provides a means for providing accurate time recordings for when an event occurred as well as for non-repudiation and integrity checking for an investigator's actions. The Transformation System uses these Proof of Actions to store a detailed audit trail of the Transformation, including the operations that were performed and the test history of the Transformation. The investigator's rationale for requesting the Transformation is also included within these Proof of Actions. The Transformation System also allows investigators to specify provenance events regarding non-digital events, such as the transportation of evidence and the storage of evidence, which results in a Proof of Action for the particular event. These Proof of Actions of a Container are stored in the Container's Proof of Action History, allowing all provenance information regarding an investigator's building blocks and how it came to be in its current form, to be easily accessible. The detailed audit trail regarding the provenance information of each Container provides reliability to an investigator's work, particularly through reconstruction by a third party, since a third party can use such an audit trail for reconstruction. However, we did not specify a format for these Proof of Actions, nor for the Proof of Action history. Further research is required to determine the optimal formats for doing so. Further research is also required to examine whether these Containers and their structure are optimal for the Chain of Findings, or whether improvements can be made.

The Chain of Findings can be seen as the glue of our research and incorporates each of the components we discussed thus far, including the Transformation System, to provide the necessary functionality and to meet the requirements we set for the Chain of Findings. The Chain of Findings includes an interface that allows investigators to more easily interact with each of its components. In addition, the Chain of Findings incorporates a new component, namely a report generator, in order to allow investigators to present their evidence and findings in a manner that best suits their needs. The Chain of Findings is also responsible for the storage and management of all Evidence and Finding Containers on behalf of the investigator. Our Chain of Findings incorporates an authentication, access control and investigation management layer, however, these aspects are not the focus of our research. Future research is required to determine the optimal approach for incorporating such aspects into the model.

In Chapter 6 we explored some additional benefits of the Chain of Findings, including how the Chain of Findings can aid in achieving Digital Forensic Readiness and in the automation of an investigator's reasoning process. In our real world example we were also able to demonstrate how the Chain of Findings can use empty Evidence Containers (called Null Containers) as a form of guideline for evidence that is needed for the investigation. While the Chain of Findings and particularly Finding Containers may be complex and hence suffer as a

result, in Chapter 6 we discussed how one can create example and template Finding Containers to train investigators and illustrate how a reasoning process can be captured. In addition, these Containers can illustrate how to correctly transform evidence, and how best to capture the certainty that an investigator's building blocks and confirmations provide for their arguments.

In order to demonstrate that the Chain of Findings is not purely theoretical, we created a prototype that served as a proof of concept. The prototype played an important role in our presentations and it allowed us to illustrate various aspects to the audience. It also allowed us to communicate the benefits of the Chain of Findings to the audience during these presentations and to highlight future work.

Future work is required to determine the most optimal approach for creating the metadata attribute class repository, the Transformation System, the various Containers and the Report Generator. We also want to investigate the optimal means for performing testing (including calibration testing) on a Transformation, and the optimal means for representing and capturing such information within the test history of a Transformation. Furthermore, we want to investigate the best means for representing actions within the Proof of Actions, particularly due to the important role they play in capturing provenance information regarding a Container. We would like to do so to ensure that Proof of Actions are well-understood and to ensure that the same approach and format is used consistently among all Transformations. The next step for future research is to perform a case study, in which our model is used to aid in a real world investigation. Such a case study will provide us with the opportunity to see the benefits and drawbacks of the model in a real-world scenario. It will also allow us to get feedback and determine where improvements of the model, and its implementation, may be required.

References

1. Adobe. (2006). *Redaction of Confidential Information in Electronic Documents.* Retrieved October 6, 2012, from http://partners.adobe.com/public/developer/en/acrobat/Redaction.pdf

2. Ahmad, A. (2002). The Forensic Chain of Evidence Model: Improving the Process of Evidence Collection in Incident Handling Procedures. *Proceedings of the 6th Pacific Asia Conference on Information Systems (PACIS 2002)* (pp. 1-5). Tokyo: Electronic version retrieved from University of Melbourne May 4, 2012. Website http://www.dis.unimelb.edu.au/staff/atif/AhmadPACIS.pdf.

3. Ahmad, A., & Ruighaver, A. B. (2003). Improved Event Logging for Security and Forensics: Developing Audit Management Infrastructure Requirements. *Proceedings of the 2003 ISOneWorld International Conference*, (pp. 1-10). Las Vegas.

4. Ashcroft, J. (2001). *Electronic Crime Scene Investigation – a Guide for First Responders.* Retrieved November 14, 2012, from National Criminal Justice Reference Service: https://www.ncjrs.gov/pdffiles1/nij/187736.pdf

5. Ben-Gal, I. (2007). Bayesian Networks. In F. Ruggeri, S. Faltin, & R. Kennet, *Encyclopaedia of Statistics in Quality & Reliability*. New Jersey: Wiley and Sons.

6. Bernard, L., Einspanier, U., Haubrock, S., Hubner, S., Kuhn, W., Lessing, R., et al. (2003). Ontologies for Intelligent Search and Semantic Translation in Spatial Data Infrastructures. *Photogrammetrie-Fernerkundung-Geoinformation , 2003* (6), pp. 451-462.

7. Brookshear, J. G. (2007). *Computer Science: An Overview* (9th ed.). Boston: Pearson Education.

8. Broucek, V., & Turner, P. (2001). Forensic Computing: Developing a Conceptual Approach for an Emerging Academic Discipline. *Armstrong, H. (ed.) 5th Australian Security Research Symposium* (pp. 55-68). Perth: Edith Cowan University.

9. Broucek, V., & Turner, P. (2004). Computer Incident Investigations: Forensic Insights on Evidence Acquisition. *U. E. Gattiker (Ed.), Proceedings of the 13th Annual EICAR Conference* (pp. 1-15). Luxembourg: EICAR.

10. Carrier, B. (2002). *Open source digital forensics tools: The legal argument (Research Report)*. Retrieved March 2, 2011, from @Stake: www.atstake.com/research/reports/acrobat/atstake_opensource_forensics.pdf.

11. Carrier, B. (2003). Defining Digital Forensic Examination and Analysis Tools Using Abstraction Layers. *International Journal of Digital Evidence , 1* (4), pp. 1-12.

12. Carrier, B., & Spafford, E. H. (2003). Getting Physical with the Digital Investigation Process. *International Journal of Digital Evidence , 2* (2), pp. 1-20.

13. Casey, E. (2002). Error, Uncertainty, and Loss in Digital Evidence. *International Journal of Digital Evidence , 1* (2).

14. Coffey, T., & Saidha, P. (1996). Non-repudiation with Mandatory Proof of Receipt. *Computer Communication Review (ACM) , 26* (1), pp. 6-17.

15. Cohen, F. (2010). *Digital Forensic Evidence Examination* (2nd ed.). California: Fred Cohen & Associates.

16. Connolly, T., & Begg, C. (2005). *Database Systems – A Practical Approach to Design, Implementation , and Management* (4th ed.). Harlow: Pearson Education.

17. Cooper, P. (2005). Speciation in the Computing Sciences : Digital Forensics as an Emerging Academic Discipline. *InfoSecCD '05 Proceedings of the 2nd Annual Conference on Information Security Curriculum Development* (pp. 19-23). New York: ACM.

18. Coppin, B. (2004). *Artificial Intelligence Illuminated.* Massachusetts: Jones & Bartlett Learning.

19. Day, J. (2008). *Patterns in Network Architecture – A Return to Fundamentals.* Boston: Pearson Education.

20. De Vel, O., Liu, N., Caelli, T., & Caetano, T. S. (2006). An Embedded Bayesian Network Hidden Markov Model for Digital Forensics. *S. Mehrotra, D.D. Zeng, & H. Chen (Eds.), Proceedings of the IEEE Conference on Intelligence and Security Informatics* (pp. 459-465). California: IEEE.

21. Deitel, P. J., & Deitel, H. M. (2007). *Java – How to Program* (7th ed.). New Jersey: Pearson Education.

22. Endicott-Popovsky, B. E., & Frincke, D. A. (2007). The Observability Calibration Test Development Framework. *Proceedings of the 2007 IEEE Workshop on Information Assurance and Security* (pp. 61-66). New York: IEEE SMC.

23. Endicott-Popovsky, B. E., Fluckiger, J. D., & Frincke, D. A. (2007). Establishing Tap Reliability in Expert Witness Testimony: Using Scenarios to Identify Calibration Needs. *Proceedings of the Second International Workshop on Systematic Approaches to Digital Forensic Engineering* (pp. 131-146). Seattle: IEEE Computer Society.

24. Endicott-Popovsky, B. E., Frincke, D. A., & Taylor, C. A. (2007). A Theoretical Framework for Organizational Network Forensic Readiness. *Journal of Computers , 2* (3), pp. 1-11.

25. Epp, S. S. (2004). *Discrete Mathematics with Applications* (3rd ed.). California: Brooks Cole.

26. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns Elements of Reusable Object-Orientated Software.* Boston: Addison-Wesley.

27. Garfinkel, S., Farrel, P., Roussev, V., & Dinolt, G. (2009). Bringing Science to Digital Forensics with Standardized Forensic Corpora. *Digital Investigation, Proceedings of the Ninth Annual DFRWS Conference* (pp. s2-s11). Montreal: Elsevier.

28. Gordon, L. A., Loeb, M. P., Lucyshyn, W., & Richards, R. (2006). *2006 CSI/FBI Computer Crime and Security Survey.* California: Computer Security Institute.

29. Grance, T., Hash, J., & Stevens, M. (2004). *Security Considerations in the Information System Development Life Cycle.* Retrieved May 7, 2012, from U.S. Department of Commerce, NIST Special Publication 800-64: http://www.itl.nist.gov/lab/bulletns/bltndec03.htm

30. Heckerman, D. (1995). *A Tutorial on Learning with Bayesian Network, Microsoft Research – Technical Report.* Retrieved April 5, 2012, from Microsoft Research: http://research.microsoft.com/pubs/69588/tr-95-06.pdf

31. Horstmann, C. (2005). *Java Concepts* (4th ed.). Massachusetts: Wiley & Sons.

32. Jansen, W., & Ayers, R. (2007). *Guidelines on Cell Phone Forensics.* Retrieved May 8, 2012, from NIST Information Technology Laboratory: http://csrc.nist.gov/publications/nistpubs/800-101/SP800-101.pdf

33. Kashyap, V., & Sheth, A. (1997). Semantic Heterogeneity in Global Information Systems: The Role of Metadata, Context and Ontologies. *M. Papazoglou and G. Schlageter (Eds.): Cooperative Information Systems: Current Trends and Directions* (pp. 139-178). Cleveland: Academic Press.

34. Kremer, S., Markowitch, S., & Zhou, J. (2002). An Intensive Survey of Fair Non Repudiation Protocols. *Computer Communications , 25* (2002), pp. 1606-1621.

35. Krishnamoorthy, C. S., & Rajeev, S. (1996). *Artificial Intelligence and Expert Systems for Engineers.* Florida: CRC Press.

36. Lasater, C. G. (2006). *Design Patterns.* Massachusetts: Jones & Bartlett Learning.

37. Lee, R., Lang, S.-D., & Stenger, K. (2009). From Digital Forensic Report to Bayesian Network Representation. *Proceedings of the 2009 IEEE International Conference on Intelligence and Security Informatics, ISI '09* (pp. 303-306). Piscataway: IEEE.

38. Luger, G. F. (2005). *Artificial Intelligence: Structures and Strategies for Complex Problem Solving* (5th ed.). Harlow: Pearson Education.

39. Lyle, J. R. (2003). NIST CFTT: Testing Disk Imaging Tools. *International Journal of Digital Evidence , 1* (4), pp. 1-10.

40. Maconachy, V., Schou, C., Ragsdale, D., & Welch, D. (2001). A Model for Information Assurance: An Integrated Approach. *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security* (pp. 306-310). New York: IEEE.

41. Manes, G. W., Watson, L., Downing, E., Barclay, A., Greer, D., & Hale, J. (2007). A Framework for Redacting Digital Information from Electronic Devices. *Proceedings on the 2007 IEEE Workshop on Information Assurance and Security* (pp. 56-60). New York: IEEE SMC.

42. Manson, D., Carlin, A., Ramos, S., Gyger, A., Kaufman, M., & Treichelt, J. (2007). Is the Open Way a Better Way? Digital Forensics Using Open Source Tools. *HICSS '07: Proceedings of the 40th Annual Hawaii International Conference on System Sciences.* Washington: IEEE Computer Society.

43. McDonald, J. T., Kim, Y. C., & Yasinsac, A. (2008). Software Issues in Digital Forensics. *SIGOPS Operating Systems Rev.(ACM) , 42* (3), pp. 29-40.

44. McGonical, J. (2011). *Reality is Broken – Why Games Make Us Better and How they Can Change the World.* London: Jonathon Cape.

45. Mohay, G. (2005). Technical Challenges and Directions for Digital Forensics. *Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering* (pp. 155-161). Washington: IEEE Computer Society.

46. Myers, B. L., Kappelman, L. A., & Prybutok, V. R. (1998). A Comprehensive Model for Assessing the Quality and Productivity of the Information Systems Function: Toward a Theory for Information Systems Assessment. *E.J. Garrity and G.L. Sanders (eds.), Information Systems Success Measurement* (pp. 94–121). Pennsylvania: IGI Publishing.

47. NIJ. (2004). Forensic Examination of Digital Evidence: A Guide for Law Enforcement. *National Institute of Justice Special Report* , Electronic version retrieved May 7, 2012 from National Criminal Justice Reference Service: http://www.ncjrs.gov/pdffiles1/nij/199408.pdf.

48. NIST. (n.d.). *Incorporating Security into the System Development Life Cycle (SDLC).* Retrieved April 30, 2012, from NIST: http://www.itl.nist.gov/lab/bulletns/bltndec03.htm

49. Noy, N. F., & McGuinness, D. L. (n.d.). *Ontology Development 101: A Guide to Creating Your First Ontology.* Retrieved April 11, 2013, from Stanford University: http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html

50. Olivier, M. S. (2009). *Information Technology Research – A Practical Guide for Computer Science and Informatics* (3rd ed.). Pretoria: Van Schaik Publishers.

51. Olivier, M. S. (2009). On Metadata Context in Database Forensics. *Digital Investigation: The International Journal of Digital Forensics & Incident Response archive , 5* (3-4), pp. 115-123.

52. Pfleeger, C., & Pfleeger, S. (2007). *Security in Computing* (4th ed.). Massachusetts: Pearson Education.

53. Pointcheval, D., & Stern, J. (2000). Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology , 13* (3), pp. 361-396.

54. Raghavan, S., Clark, A., & Mohay, G. (2009). FIA: An Open Forensic Integration Architecture for Composing Digital Evidence. *Proceedings of the 2nd International ICST Conference on Forensic Applications and Techniques in Telecommunications, Information, and Multimedia (eForensics 10), LNICS 1867-8211. 8*, pp. 83-94. Adelaide: Springer.

55. Renaud, K., & De Angeli, A. (2009). Visual Passwords: Cure-all or Snake-oil?'. *Communications of the ACM , 52* (12), pp. 135-140.

56. Rigaux, P., Scoll, M., & Voisard, A. (2002). *Spatial Databases with Application to GIS.* San Francisco: Morgan Kaufmann Publishers.

57. Rob, P., & Coronel, C. (2007). *Database Systems Design, Implementation and Management* (7th ed.). Massachusetts: Thompson Course Technology.

58. Rowlingson, R. (2004). A Ten Step Process for Forensic Readiness. *International Journal of Digital Evidence , 2* (3), pp. 1-28.

59. Schatz, B., & Clark, A. (2006). An Open Architecture for Digital Evidence Integration. *Proceedings of the 2006 AUSCERT R&D Stream* (pp. 15-29). Queensland: Queensland University of Technology.

60. Sebesta, R. (2008). *Programming Languages* (8th ed.). Massachusetts: Pearson Education.

61. Sharp, H., Rogers, Y., & Preece, J. (2007). *Interaction Design – Beyond Human Computer Interaction* (2nd ed.). Chichester: Wiley & Sons.

62. Sheth, A., & Kashyap, V. (1993). So far (schematically) yet so near (semantically). *Proceedings of the IFIP WG 2.6 Database Semantics Conference on Interoperable Database Systems (DS-5)* (pp. 283-312). Amsterdam: North-Holland Publishing Co.

63. Sheth, A., & Kashyap, V. (1996). Media-independent Correlation of Information: What? How? *Proceedings of First IEEE Metadata Conference.* Maryland: IEEE.

64. Tan, J. (2001). *Forensic Readiness.* Retrieved May 5, 2012, from @Stake: http://www.atstake.com/research/reports/acrobat/atstake_forensic_readiness.pdf

65. Terry, P. (2005). *Compiling with C# and JAVA.* Harlow: Pearson Education.

66. *The NTP FAQ and HOWTO.* (2006). Retrieved March 23, 2012, from http://www.ntp.org/ntpfaq/NTP-a-faq.htm

67. Turner, P. (2005). Digital Provenance - Interpretation, Verification and Corroboration. *Digital Investigation , 2* (2), 45-49.

68. Turner, P. (2005). Unification of Digital Evidence from Disparate Sources (Digital Evidence Bags). *Digital Investigation , 2* (3), pp. 223-228.

69. Turner, P. (2006). Selective and Intelligent Imaging using Digital Evidence Bags. *The Proceedings of the 6th Annual Digital Forensic Research Workshop (DFRWS '06). 3 (Supplement 1)*, pp. 59-64. Amsterdam: Elsevier Science Publishers.

70. Wache, H., Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., et al. (2001). Ontology-Based Integration of Information – a Survey of Existing Approaches. *Proceedings of the IJCAI Workshop on Ontologies and Information Sharing*, (pp. 108–117). Washington.

71. Wagner, D., & Soto, P. (2002). Mimicry Attacks on Host-Based Intrusion Detection Systems. *CCS '02 Proceedings of the 9th ACM Conference on Computer and Communications Security* (pp. 255-264). New York: ACM.

72. Wang, W., & Daniels, T. E. (2005). Building Evidence Graphs for Network Forensics Analysis. *Proceedings of the 21st Annual Computer Security Applications Conference (ACSAC 2005)* (pp. 256-266). Arizona: IEEE ACSAC.

73. Wilkinson, S., & ACPO. (2003). *Good Practice Guide for Computer-based Electronic Evidence.* Retrieved May 1, 2012, from www.nhtcu.org/ACPO%20Guide%20v3.0.pdf

74. Wolfe-Wilson, J., & Wolfe, H. B. (2003). Management Strategies for Implementing Forensic Security Measures. *Information Security Technical Report , 8* (2), pp. 55-64.

75. Yasinsac, A., & Manzano, Y. (2001). Policies to Enhance Computer and Network Forensics. *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security* (pp. 289-295). New York: IEEE.

76. Ye, N., Giordano, J., Feldman, J., & Zhong, Q. (1998). Information Fusion Techniques for Network Intrusion Detection. *IEEE Information Technology Conference, Information Environment For The Future (1998)* (pp. 117-120). New York: IEEE.

77. Zheng, Y. (1997). Digital Signcryption or How to Achieve Cost (Signature & Encryption) <<Cost (Signature) + Cost (Encryption). *Proceedings Crypto '97, LNCS 1294* (pp. 165-179). California: Springer.