Digital Evidence Bags

Philip Bryan Turner (BSc. Hons, MIET)

This thesis is submitted in partial fulfilment of the requirements of the award of Doctor of Philosophy

Oxford Brookes University
Wheatley Campus
Oxford
OX33 1HX

June 2008

Abstract

This thesis analyses the traditional approach and methodology used to conduct digital forensic information capture, analysis and investigation. The predominant toolsets and utilities that are used and the features that they provide are reviewed. This is used to highlight the difficulties that are encountered due to both technological advances and the methodologies employed. It is suggested that these difficulties are compounded by the archaic methods and proprietary formats that are used.

An alternative framework for the capture and storage of information used in digital forensics is defined named the 'Digital Evidence Bag' (DEB). A DEB is a universal extensible container for the storage of digital information acquired from any digital source. The format of which can be manipulated to meet the requirements of the particular information that is to be stored. The format definition is extensible thereby allowing it to encompass new sources of data, cryptographic and compression algorithms and protocols as developed, whilst also providing the flexibility for some degree of backwards compatibility as the format develops.

The DEB framework utilises terminology to define its various components that are analogous with evidence bags, tags and seals used for traditional physical evidence storage and continuity. This is crucial for ensuring that the functionality provided by each component is comprehensible by the general public, judiciary and law enforcement personnel without detracting or obscuring the evidential information contained within.

Furthermore, information can be acquired from a dynamic or more traditional static environment and from a disparate range of digital devices. The flexibility of the DEB framework permits selective and/or intelligent acquisition methods to be employed together with enhanced provenance and continuity audit trails to be recorded. Evidential integrity is assured using accepted cryptographic techniques and algorithms.

The DEB framework is implemented in a number of tool demonstrators and applied to a number of typical scenarios that illustrate the flexibility of the DEB framework and format.

The DEB framework has also formed the basis of a patent application.

Acknowledgements

I would like to thank my employer QinetiQ for sponsoring this PhD, and all the members of the Digital Investigation Services team at QinetiQ for their tolerance and patience in listening to me endlessly suggesting how Digital Evidence Bags could be the answer to the problems that are encountered during their investigative work.

I would also like to express my gratitude to Professor David Duce and Dr. Faye Mitchell for steering me through the necessary procedural requirements of Oxford Brookes and the PhD process. Additionally, for encouraging me to publish, present and for providing an independent validation, dimension and view of my ideas.

The following additional people have kindly reviewed my work providing invaluable constructive feedback, comments and support:

Gerry Masters – The most talented programmer, paper reviewer and supplier of the wittiest programming comments you could ever wish for.

Dr. Ian & Cathy Taylor – For reviewing papers and convincing me that a PhD was worthwhile undertaking.

Finally, I would like to thank my friends and family for their full support and encouragement to undertake this work and see it through.

I dedicate this with love to:

Sandy, Kerry and Sam

Table of Contents

Abstract ii
Acknowledgements iv
Table of Contents v
List of Tables vii
List of figures viii
1. Introduction 1
1.1 Digital Forensics – The problem 1
1.2 Research Objectives 3
1.3 Original Contribution 3
1.4 Thesis Outline 4
2 Digital Forensics Background & Related Work 6
2.1 The Influential Stakeholders of the Digital Forensic Process 6
2.1.1 Law Enforcement 6
2.1.2 Academia 8
2.1.3 Legislation 9
2.2 Overview of the Digital Investigation Process 10
2.2.1 Digital Imaging 13
2.2.2 Digital Examination 21
2.3 Problems with current methods 26
2.4 Imaging and Analysis Tool Requirements 30
2.5 Scope of Thesis 33
3 Digital Evidence Bags (DEB) 34
3.1 Traditional Evidence Storage – Bags, Tags and Seals 34
3.1.1 Characteristics of Bags, Tags and Seals 37
3.2 DEB Concept 37
3.3 DEB Components 39
3.3.1 DEB Tag 40
3.3.2 DEB Index 44
3.3.3 DEB Bag 44
3.4 DEB Characteristics 44
3.4.1 Provenance 44
3.4.2 Confidentiality 48
3.4.3 Integrity 48
3.4.4 Availability 49
3.4.5 Continuity 50
3.4.6 Process 51
3.4.7 Time 51
3.5 DEB Syntax Definition 52
3.5.1 Tag File – General Information 53
3.5.2 Index File 63
3.5.3 Bag File 64
3.6 DEB API Implementation 64
3.6.1 DEB Creation functions 66
3.6.2 DEB Access functions 68
3.6.3 Alternatives to using the DEB API 71
4 DEB Toolkit Implementation and Framework Demonstrators 73

4.1 DEB Creation Tool Demonstrator 74
4.2 DEB Access/Viewer Tool Demonstrator 78
4.3 Enhancing Current Applications - DEB Command Line Wrapper 82
5 Digital Investigation Scenarios & Case Studies 86
5.1 Device / Media Imaging 86
5.1.1 DEB Device Imager application 87
5.1.2 Selective Imaging 88
5.1.3 Intelligent Imaging 90
5.1.4 Selective & Intelligent Imaging using Digital Evidence Bags 91
5.2 Incident Response – System Management 93
5.2.1 Incident Response - Common Tools 94
5.2.2 System and Application operation log files 95
5.2.3 System Administration Tools - Command line utilities 97
5.3 Magnetic Stripe Card Cloning devices 99
6 Comparison of DEB framework with other Evidence Storage Formats 104
6.1 Digital Forensic Image Storage Formats 104
6.1.1 Advanced Forensic Format (AFF) 104
6.1.2 Expert Witness – EnCase - SMART 104
6.1.3 EnCase 104
6.1.4 SMART 105
6.1.5 Generic Forensic Zip (Gfzip) 105
6.1.6 ProDiscover 105
6.2 Format Comparison 106
6.3 DEB Associated Work 111
7 Conclusions & Recommendations for Future Work 114
7.1 Conclusions 114
7.2 Recommendations for Further Work 117
8 References 120
8.1 Conferences Attended and Presentations Given 127
9 Appendix A – Published papers 130
9.1 Digital provenance – interpretation, verification and corroboration 131
9.2 Unification of digital evidence from disparate sources 136
9.3 Digital Evidence Provenance and Continuity using DEBs 142
9.4 Selective and Intelligent imaging using digital evidence bags 146
9.5 Applying a forensic approach to incident response, network investigation and system administration using Digital Evidence Bags 152
9.6 Standardizing Digital Evidence Storage - The Common Digital Evidence Storage Format Working Group 158
9.7 Forensic data recovery and examination of magnetic swipe card cloning devices 160
9.8 Digital Forensics : Challenges and Opportunities 167
10 Appendix B - Common Digital Evidence Storage Format (CDESF) working group – DFRWS 173
10.1 Appendix B-2 - CDESF – Format Survey Information 174
10.2 Appendix B-3 - CDESF – Survey of Disk Image Storage Formats 176
11 Appendix C - Patent application 194

List of Tables

Table 1 - Digital Forensics Issues & Requirements Correlation.....32
Table 2 - ACID Properties & Requirements Correlation.....43
Table 3 - Format Comparison Summary.....110
Table 4 - DEB Framework & Requirements Correlation 117

List of figures

Figure 1 - DOS EnCase Acquisition screen 1 16
Figure 2 - DOS EnCase Acquisition screen 2 17
Figure 3 - Windows EnCase Acquisition screen 1 18
Figure 4 - Windows EnCase Acquisition screen 2 19
Figure 5 - EnCase Evidence File structure 19
Figure 6 - FTK Imager Acquisition screen 20
Figure 7 - EnCase analysis tool - screenshot 23
Figure 8 - FTK analysis tool - screenshot 25
Figure 9 - Transparent evidence bag label 35
Figure 10 - Evidence tag 36
Figure 11 - Tamper-proof seal - 1 36
Figure 12 - Tamper-proof seal - 2 37
Figure 13 - Digital Evidence Bag basic structure 39
Figure 14 - Tag structure 40
Figure 15 - Detailed DEB diagram 53
Figure 16 - DEB Writer API demonstrator 75
Figure 17 - DEB Writer Flow diagram 76
Figure 18 - DEB Viewer demonstrator 79
Figure 19 - Hexadecimal display of DEB Viewer demonstrator 81
Figure 20 - Native View output example 82
Figure 21 - DEB Information segregation 84
Figure 22 - DEB Command Line Application Wrapper screenshot 85
Figure 23 - DEB structure diagram 87
Figure 24 - DEB Imager screenshot 88
Figure 25 - Disparate digital device capture into a single DEB 91
Figure 26 - DEB Structure for selective / intelligent imaging 92
Figure 27 - DEB Selective Image structure 93
Figure 28 - MSR-500M (Mini –123) Magnetic Swipe Card Reader 101
Figure 29 - MSR206 Magnetic Swipe Encoder 101
Figure 30 - Digital Evidence Bag schematic of skimmer information 102

1. Introduction

1.1 Digital Forensics – The problem

Digital / computer forensics is one of the more immature and relatively new forms of forensic science. In addition to that the technology to which digital forensic techniques and a scientific rigorous approach is applied are also changing at a phenomenally rapid rate.

In contrast to digital forensics in which the processes and procedures are not well defined, the 'wet' forms of forensic science such as dactylography (fingerprints) (*Thinkquest 2004*), (*German 2007*) and DNA (*Butler 2005*) are much more formally established (*Pyrek 2007*). The processes and procedures are well defined (*DOJ 2003*) and the raw material to which those processes and procedures are applied are not changing (no more than the human DNA microbiological makeup is evolving).

The rate of technological change makes it even more vitally important to record the processes, methods, tools and actions taken on digital evidence, and the work described in this thesis proposes a solution that is capable of recording these. In addition to that the provenance, chain of custody and integrity of the digital evidence should be maintained just as it would be with more traditional forensic sciences. This however is not currently the case in the digital environment. The digital world is very good at providing mechanisms to help assure data integrity but is currently very poor at providing mechanisms to record provenance, processes, methods, tools and actions. To accomplish this, the investigator / examiner has to resort to manually recording these aspects of the examination. Although not difficult to perform it does rely on a rigorous and studious approach on the part of the investigator but does not allow the integrity to be independently verified.

Another dimension that compounds these problems in the digital forensic field is the sheer quantity of information that can be stored on a digital device and the rate at which this is increasing (*Thompson & Best 2000*). This incurs additional backup overheads and increases the time to thoroughly parse and examine all of the digital information collected as part of an investigation. The procedure of capturing all

available data from, for instance, a computer hard disk drive and examining the clone or image file has not changed since computer forensics was first undertaken in the early 1980's.

Digital forensics is normally seen as taking a static approach to digital investigations. however with 'live' investigations and data capture from a system that is up and running the methods and tools that are used are even less well defined, and any data obtained during this live acquisition is rarely compatible in format to that obtained during a static acquisition. This means different toolsets are used and the investigator has to be very technically astute; to the extent that we are even seeing some investigators specialising in 'network' or 'live' digital forensics. Indeed performing a basic live data capture will soon be the norm as it becomes more recognised that capturing the contents of memory, a process list, or list of network connections from the dynamic environment may well aid the investigation. This contains information that would be advantageous to have but is totally lost once a static approach is undertaken. Furthermore, if a proactive approach is adopted when the investigator arrives at a crime scene and is faced with a live system, then the recognition of encrypted volumes and such schemes may well allow the investigation to proceed without additional delays and costs that may be incurred to circumvent such protection at a later date. In extreme cases these methods may well render the seized digital media totally worthless from an investigatory perspective.

This thesis predominantly discusses the investigation of Microsoft operating systems and filing systems as these are the most commonly encountered by practitioners and documented (*Steel 2006*) (*Carrier 2005*) (*Sammes & Jenkinson 2000*). The most widely used commercial forensic tools are also built to handle these systems. The principles and practices discussed are equally applicable to other operating systems (e.g. Unix, Apple Machintosh, etc.) and filing systems (e.g. EXT2, EXT3, HFS, etc.).

1.2 Research Objectives

The objective of this work is to address the problems stated in section 1.1 and produce a framework that is capable of recording the provenance, processes, methods, tools and actions taken by the investigator.

Such a framework should permit an efficient and scaleable storage of digital information captured and should allow information to be captured to evidential standards from both static and live environments, safe in the knowledge that once that information has been seized not only the integrity of the information is maintained but provenance is also recorded.

A framework that can successfully achieve these objectives may even lead to the possibility of new forensic investigation methodologies being developed.

A by-product of this would be to harmonise the digital forensic techniques and tools that can be applied to the investigation of an ever increasing range of disparate digital devices.

1.3 Original Contribution

A critical analysis of current methodologies and problems encountered in the digital forensic arena is performed (section 2.1 to 2.3). This is used to identify a set of requirements that digital forensic imaging and analysis tools should possess (section 2.4). As a result of this an original solution has been developed which addresses many of these issues. The solution as described in this thesis is named the Digital Evidence Bag (DEB), section 3.

A DEB is a universal storage container for digital information, which may be used as evidence, and is collected from a disparate range of digital devices in either a static or live environment. The process that is undertaken to capture the digital information is also recorded.

A sufficiently detailed definition has been formulated to permit a patent application to be filed defining a scaleable and novel solution that is capable of recording the

provenance, processes, methods, tools and actions taken by the investigator during the course of digital evidence acquisition and analysis.

1.4 Thesis Outline

This thesis examines and addresses a number of areas not previously considered or at least documented in the digital forensics field. The chapters are organised as follows:

Chapter 2 discusses the influential stakeholders in the digital forensic arena and then presents an overview of the current digital investigation process, methodology and tools. A number of problems with current practice are identified that leads to the definition of requirements for a digital evidence imaging system.

In Chapter 3 the importance of provenance is considered, and attributes that characterise 'good' provenance are defined. This is an area of digital evidence that is often overlooked, but is key to providing reliable and repeatable evidence. The abstract DEB framework is presented together with a description of the various components that comprise a DEB. The sequence that should be adopted when a DEB is created together with the DEB access protocol used during the course of an investigation is also defined. A number of DEB tools were created in order to test the DEB framework and examine the functionality and usability of the system.

Chapter 4 discusses the applications that have been written that are capable of creating a DEB, both in a traditional static environment – an imaging tool (DEB imager) and also able to capture information from a real-time incident response environment (DEB Application Wrapper). Additionally, a tool was also created that was capable of viewing the contents of a DEB (DEB Viewer) in order to demonstrate and test the protocols that are undertaken when the information contained in a DEB is examined and analysed during the course of the investigation.

In Chapter 5, the DEB framework is applied to a number of possible digital investigation scenarios. This demonstrates how the DEB framework can be implemented in current scenarios and illustrates how new intelligent and selective imaging practices can be accommodated.

A comparison of the DEB framework with other digital evidence storage formats is conducted in Chapter 6. This highlights the range of features, flexibility and extensibility of the DEB approach. The final chapter, Chapter 7, draws the conclusions of the research and gives some recommendations for further work.

2 Digital Forensics Background & Related Work

2.1 The Influential Stakeholders of the Digital Forensic Process

As digital forensics is a relatively new branch of forensic science the framework and processes that are undertaken throughout the lifecycle of an investigation have only recently been discussed and documented (*Kovacich & Jones 2006*) (*Casey 2004*). As such, there are no internationally defined standards in this area. The main influencing communities are digital forensic practitioners, academia and legislative. The practitioner skill and knowledge base is driven from the Law Enforcement community that regularly attends crime scenes and performs search and seizure operations in order to secure digital evidence. The academia dimension is driven by the requirement to gain an understanding of the overall process and produce innovative solutions to address the technical challenges which digital evidence can encapsulate. The other aspect that greatly influences the digital investigation process is the law and legislation that is used to define offences and prosecute offenders. These areas will now be discussed in further detail and from this a number of requirements are identified.

2.1.1 Law Enforcement

From the law enforcement perspective, certainly within the UK, the Association of Chief Police Officers (ACPO) provides advice and guidance to UK law enforcement. In the area of digital forensics ACPO have developed a 'Good Practice Guide for Computer Based Electronic Evidence' (*ACPO 2003*). This guide as well as providing some background technical advice defines four main principles that should be applied when investigating any computer or digital system. These are :-

Principle 1: No action taken by law enforcement agencies or their agents should change data held on a computer or storage media which may subsequently be relied upon in court.

This gives rise to the requirement to provide a means to be able to verify that the data presented at court is identical to that acquired from the original media. This assurance is usually achieved in the digital forensic arena using cryptographic integrity checks.

Principle 2: In exceptional circumstances, where a person finds it necessary to access original data held on a computer or on storage media, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions.

The ability to be able to explain technical aspects of an investigation in a way that can be easily understood in a court, tribunal or even in a case study for training purposes is an essential skill for the law enforcement practitioner.

Principle 3: An audit trail or other record of all processes applied to computer based electronic evidence should be created and preserved. An independent third party should be able to examine those processes and achieve the same result.

This identifies another requirement; to record an audit trail of actions and applications (including version number) that could be used to perform independent verification of examination results and findings. This also aids the process of dual-tool verification and the ability to identify discrepancies between various different toolsets and alternate versions of the same tool.

Principle 4: The person in charge of the investigation (the case officer) has overall responsibility for ensuring that the law and these principles are adhered to.

The overall premise of these principles is that digital evidence is treated as if it were paper based … '*doctrine of documentary evidence may be explained thus: the onus is on the prosecution to show to the court that the evidence produced is no more and no less now than when it was first taken into the possession of the police.*'

Whilst these principles are very broad and not specific to any particular make, model or type of device they provide the premise, cornerstone and requirements of how all

systems should be handled if credible evidence is to be secured, documented, managed and processed.

2.1.2 Academia

Prior to 2001 there was little collaboration between academia and the 'real-world' experiences from practitioners and the legal community. This problem was identified and culminated in the creation of the Digital Forensic Research Workshop (DFRWS) supported by the US DoD Air Force Research Laboratory (AFRL).

The initial goal of the DFRWS gathering was to unite military, civilian, law enforcement and research professionals to define a framework for digital forensic science. This has resulted in a more formal understanding of the digital forensic process (*DFRWS 2001*) and the phases that are undertaken as part of the investigative process. The following phases and associated techniques or methods belonging to each phase were identified:

- Identification – Event/Crime Detection, Resolve Signature, Profile Detection, Anomalous Detection, Complaints, System Monitoring, Audit Analysis;

- Preservation – Case Management, Imaging Technologies, Chain of Custody, Time Synchronisation;

- Collection – Preservation, Approved Methods, Approved Software, Approved Hardware, Legal Authority, Lossless Compression, Sampling, Data Reduction, Recovery Techniques;

- Examination – Preservation, Traceability, Validation Techniques, Filtering Techniques, Pattern Matching, Hidden Data Discovery, Hidden Data Extraction;

- Analysis – Preservation, Traceability, Statistical, Protocols, Data Mining, Timeline, Link, Spatial;

- Presentation – Documentation, Expert Testimony, Clarification, Mission Impact Statement, Recommended Countermeasure, Statistical Interpretation:

- Decision.

Within the DFRWS framework, an investigation is very much seen as a linear process. However, certainly within the examination and analysis phases iterative tasks would be undertaken.

The techniques and methods assigned to each of the phases of an investigation also highlight additional requirements some of which are considered mandatory e.g. *Chain of Custody, Approved Methods, Preservation, Traceability*, whilst others appear optional e.g. *Approved Software, Validation Techniques, Timeline*. But aren't they all important? Who decides? Perhaps it is legislation or a general lack of understanding of what is technically possible and it is assumed that all methods and techniques are applied.

2.1.3 Legislation

In modern society the use of information technology (IT) in our lives has become evermore pervasive. This has resulted in this technology being used for an increasing number of everyday tasks and just as easily for malicious or illegal activity. The material obtained from digital technology is usually circumstantial evidence that is used to support a case. Evidence of fact is much more difficult to obtain from digital devices as without other corroborating information, it is usually very hard to prove that a particular suspect was the person who used that device or computer at the time the offence was committed.

Evidence from IT is commonly used to support the prosecution of 'traditional' crime cases. These are often crimes in which the digital technology was not even invented when that legislation was passed. For example, the Criminal Law Act (*TSO 1967*), Homicide Act (*TSO 1957*) or Offences Against the Person Act (*TSO 1861*) may be used to prosecute a murder. However, it is quite common to examine a suspect's computer to try and determine if the suspect had any connection with the victim or if

the suspect was trying to research potential ways of committing the crime without being detected.

Another example of traditional legislation being used is the Theft Act (*TSO 1968*), which is to prosecute for theft or 'going equipped to steal'. This could involve technology if a suspect was found to be carrying a credit card skimmer. This is the modern equivalent of being found in possession of lock picking equipment.

Furthermore, even the development of information technology specific based legislation often lags behind the technological advances. This is typified in the UK by the Computer Misuse Act (*TSO 1990*) which was enacted in 1990. This introduced offences such as unauthorized access to a computer, unauthorized access to a computer with the intent to commit or facilitate the commission of further offences and unauthorized modification of computer material. This was created prior to the proliferation of the World Wide Web and does not easily cover denial of service (DoS) attacks. Amendments to this legislation in 2006 in the Police and Justice Bill, Part 5, Sections 35 to 38 (*TSO 2006*) attempt to address this type of anomaly (Section 36). However, this has raised concerns that it may impede legitimate computer security industry practices with regard to handling security vulnerability information about malicious programs.

The way the judiciary keeps legislation more up-to-date and directed to specific offences is with the use of case law. For example, when prosecuting an offence of 'possession' or 'distribution' of indecent images, the Crown Prosecution Service (CPS) within the UK issued guidance based on a judge's ruling during a case, R v Oliver-Hartrey (*EWCA 2002*). This gave sentencing guidelines that resulted in the production and classification requirements (copine levels) defined in 'Practical Advice on Investigating Indecent Images of Children on the Internet' (*ACPO 2005*) required when producing evidence that was to be used in this type of case.

2.2 Overview of the Digital Investigation Process

Irrespective of the type of investigation being undertaken the process that is performed to *Preserve*, *Collect* and *Examine* digital evidence does not change.

Furthermore the specific methodology has not changed since computer forensics was first undertaken in the 1980s.

This is best demonstrated with a simple scenario:

1. A warrant is executed and crime scene attended.

2. A computer is identified at the scene and a law enforcement officer seizes the computer. The officer at the scene puts the computer in a polythene bag and secures the bag with a tamper proof seal to which is attached a tag that is duly completed with the following information:

   case identifier;
   exhibit identifier – typically the initials of the seizing officer;
   the date and time the computer was seized;
   the location the computer was seized from – e.g the address and location within a building;
   a description of the item seized;
   the name of the officer;
   the signature of the officer.

   Also at the scene, an exhibits record would be completed logging all exhibits recovered at the address. The record of the computer exhibit would be recorded in the exhibits record.

3. The officer seizing the computer would also be required to produce a witness statement which would briefly include the following information:

   Who they were – full name and position / grade or rank;
   A summary of their professional qualifications, knowledge, skills and experience;
   A list of the items seized (description) and their exhibit identifier and seal number.

4. The exhibit would then be transported to a laboratory where the property would be booked into the property store. When the item is subsequently withdrawn from the evidence store for forensic examination the entry in the property store log would be updated to reflect who, when, where and why the exhibit was withdrawn from the store.

5. The examination would typically be conducted by a digital forensic specialist who would create a set of case specific contemporaneous notes that record all actions and processes undertaken on the exhibit whilst in the control of the examiner. These contemporaneous notes and any other unused material can be disclosed to the defence by request.

6. The digital forensic specialist would typically record the following information in their contemporaneous notes:

   The exhibit reference number;
   The exhibit seal number;
   The date and time the exhibit came into their possession;
   A photograph may be taken of the exhibit as received;
   The seal on the exhibit would be broken and the contents examined. The contents of the exhibit bag would be recorded.

In this example the make, model and serial number of the computer would be noted together with a record of any damage and other details of the exhibit. For a computer an external examination would involve recording what connectivity capability or visible storage devices the system contained. The topographical layout of the features would be noted and usually photographed and removable media devices examined for the presence of media. An internal examination would then be conducted; this usually involves a certain amount of disassembly of the unit, and again suitable photographs showing the layout and configuration of the system would be taken. The internal storage media, usually a hard disk drive would be removed and its make, model, serial number and capacity noted and photographed. The removed media would then be connected to a write protection/blocking device in a laboratory imaging system. The write blocking device prevents inadvertent writing to the evidential media.

2.2.1 Digital Imaging

Digital imaging is the cornerstone of the digital forensic process, and has not changed since computer forensic investigations were first undertaken some 25 years or so ago with tools like Disk Image Backup System (DIBS) or VOGON's imagers Simage, LDi, SDi and associated analysis tools. Digital imaging is operating system and filing system independent, whereas digital examination and analysis tools discussed in section 2.2.2 have to be able to interpret and present the information captured. The imaging tools perform a basic but dumb process. The only enhancement made to this process has been the addition of hardware write-blocking devices that permit a wide and varied range of interfaces to be safely connected to an imaging system.

An imaging program is used to obtain a *full bit level image* of the source device; for the purposes of imaging discussed in this thesis 'bit level' is defined as the lowest level of data available to the imaging tool using the standard media interface. Imaging is not to be confused with 'copying' or 'backup' that is usually used to describe the capture process of the 'live' logical file structure contained on the device. The captured image contains all live and unused space. Usually a cryptographic fingerprint (hash) is then calculated over the data so that its integrity can be checked and assured at a later date.

All subsequent examination, investigation or restoration of this image is then performed after the image is verified as being a true copy by verification of the cryptographic fingerprint. There is no consistent approach to bad block handling when there is difficulty in reading the source media. Some imagers record the location of bad blocks while others ignore them. Similarly, some imagers write a 'null' filled block to the image when a bad block is encountered while others mark the block with an error signature. The method chosen obviously affects the image fingerprint and this could lead to difficulties in verifying the outputs of different tools.

The fingerprint of the captured data should be recorded and a backup of the image file should be made; usually to write-once media and produced as the primary evidence obtained from the device. Again there is no consistency about where or how the

fingerprint is recorded. Some applications append the fingerprint to the end of the image data whilst others display the value for the investigator to record manually.

Ideally a standard approach to imaging would be defined and adopted. However there is currently no International Standards body that is trying to achieve this. The Common Digital Evidence Storage Format (CDESF) working group of the Digital Forensic Research Workshop (DFRWS) was created in an attempt to tackle this issue (*CDESF 2006*). This resulted in a survey of disk imaging tools, see Appendix B (*DFRWS 2006*). The aim of this study was to identify the common features, limitations and tool support that were currently provided for the various formats.

The predominant formats and imaging tools are now briefly described together with a description of the variants of those tools that can be found, and the various metadata that may be stored with each.

2.2.1.1 Raw Imaging Format

The 'raw' format is the most basic format that is used in digital forensics. It comprises of a bit level image of the source media and no associated metadata, hence it is the most simple to generate and because of this it is the most widely supported. It was not conceived as a forensic format but just as a means of storing a full copy of the original media.

Tools and utilities exist on many operating systems in order to obtain a raw image. The most popular utility to obtain a raw image is that provided by the Unix 'dd' command that has been incorporated into Unix for many years. It is a very flexible tool that can be used to copy, clone and convert information from various block devices. The 'dd' utility can also be used in pipes so that its input and output can be fed from and to other utilities.

This flexibility allows 'dd' to be used for the forensic acquisition of any block based device to a raw image file. Although the utility is very flexible, many forensic examiners avoid its use because of the complexity that the flexibility brings.

In addition, 'dd' does not allow any metadata about the case to be collected at the time the acquisition commences. Furthermore, no integrity check (hash) is automatically generated and stored by the utility, although a hash can be generated by piping the 'dd' output through an 'md5sum' or similar hash function. When bad blocks are encountered using this utility the output image is usually filled with the same length of 'null' filled data. No provision is made to record the location of the bad blocks.

Variants of the 'dd' utility exist (*DOD 2007*) that automatically store metadata and generate integrity hashes, however the outputs from these variants are not widely supported or interpreted by other forensic examination and analysis tools.

The output from the 'dd' utility is a plain (raw) bit stream image. This is currently the only universal interchange format that all forensic tools and utilities can accept as input.

2.2.1.2 EnCase Imaging

EnCase is a forensic tool produced by Guidance Software Inc. The functionality provided by the EnCase toolset covers the Collection, Examination and Analysis phases of the DFRWS framework.

For the collection or acquisition of digital evidence three applications are available. MSDOS (known as EN) and Linux (known as Linen) based utilities allow image files (evidence files in EnCase terminology) to be created from source disk media. The functionality provided by the MSDOS and Linux versions is identical, therefore only the MSDOS version is described in this thesis. The third utility is a Microsoft Windows application used to both acquire the evidence and examine its contents. The MSDOS and Linux versions are still used on occasions due to the inability of certain devices to be accessible within the Microsoft Windows environment.

2.2.1.2.1 EnCase MSDOS Evidence Acquisition

When an acquisition is undertaken using this utility the application displays all the physical disks and logical volumes that can be found on the attached system and a software 'lock' is applied to them. The operator has to 'unlock' the target volume that

is to be used to store the image files. Use of write blocking devices with the MSDOS and Linux variants of the EnCase evidence acquisition tools is optional as these operating systems are trusted not to write to the source media. This relies on the user not making an error in selecting the output / destination media and applying the software lock appropriately.

Once the evidence and target drives are identified the following information is requested from the operator that will be embedded in the header of the image files:

- Case Number;
- Examiner [mandatory];
- Evidence Number [mandatory];
- Unique Description [mandatory];
- Current Time [mandatory];
- Notes.

Figure 1 below shows a screen capture of the DOS EnCase acquisition process once imaging is commenced.

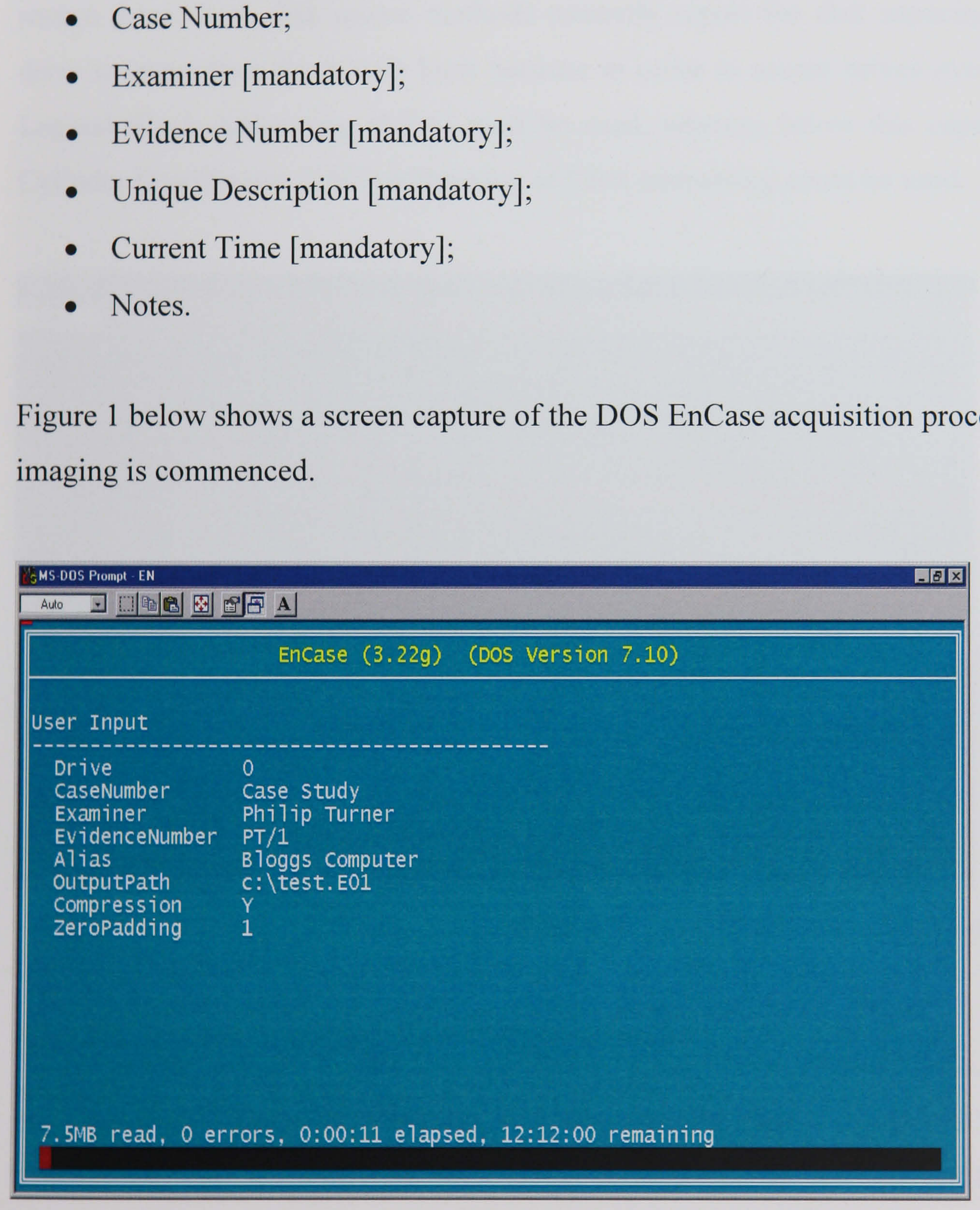

Figure 1 - DOS EnCase Acquisition screen 1

Two modes of disk access are permitted using the MSDOS acquisition method. They are 'BIOS' access and 'Direct' disk access. The BIOS method uses Int13 disk access calls to read the source media. The Direct ATA method bypasses the BIOS access method (*Microsoft 2007a*) and issues disk ATA commands (*ANSI 2005*) directly to read the source media. The disk access method is important when acquiring hard disk drives smaller than 8.5GB in capacity, as it is possible for BIOS access modes to fail to 'see' the whole hard drive. Typically the size of hard drive reported by the BIOS access method would be one cylinder less than the actual number of cylinders on the source drive. Both disk access methods correctly report the disk capacity when the drive is larger than the 8.5GB limit because in order to access drives over this limit Logical Block Addressing (LBA) must be used, whereas below this capacity either Cylinder Head Sector (CHS) addressing or LBA addressing could be used.

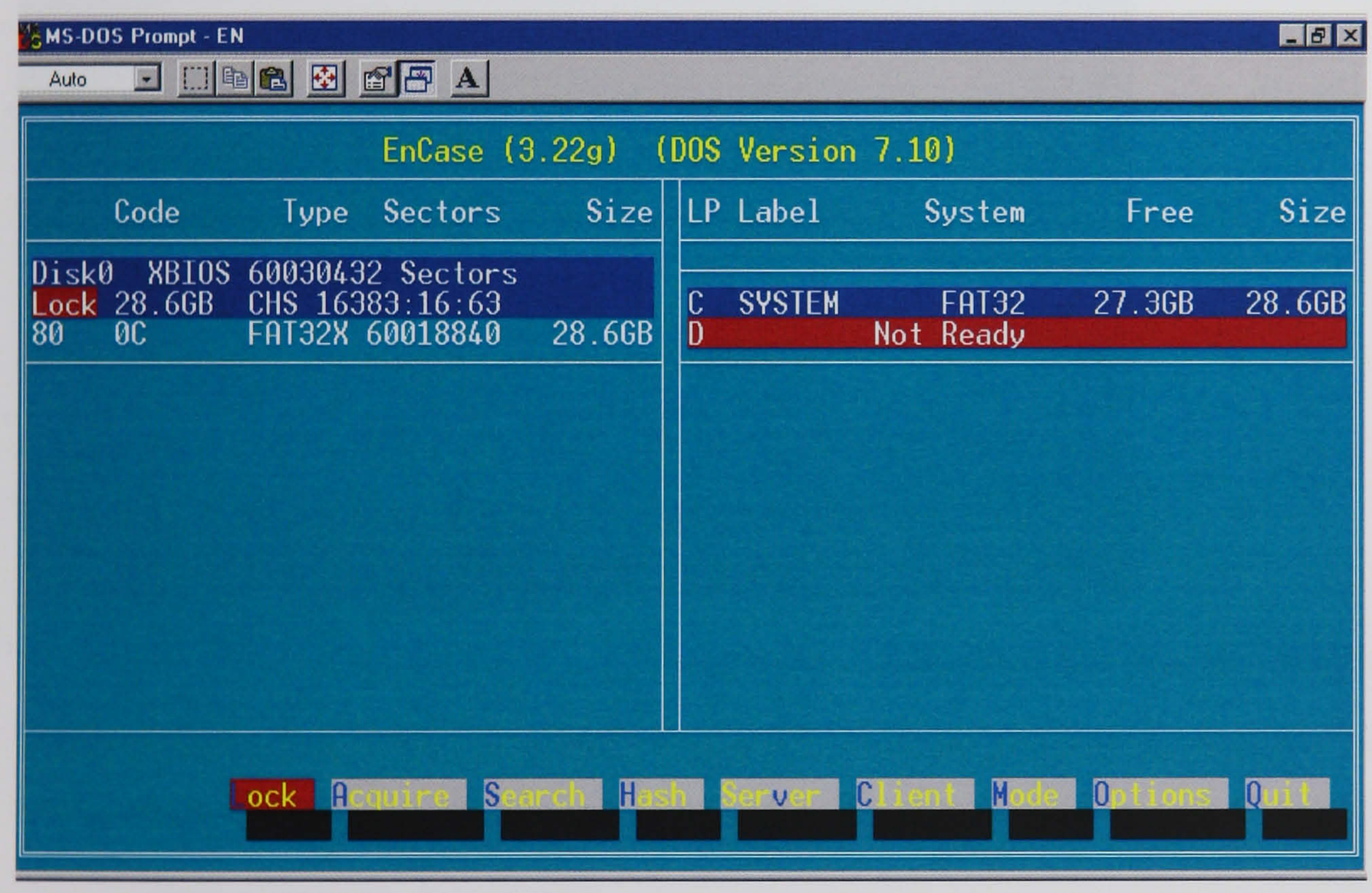

Figure 2 - DOS EnCase Acquisition screen 2

2.2.1.2.2 EnCase Windows Evidence Acquisition

Acquisition on Microsoft Windows operating systems should always be performed with the addition of a hardware write blocking device connected between the suspect drive and the acquisition system. This is done to prevent Windows accessing and changing data and file timestamps on the evidential drive (*Microsoft 2007b*).

The information requested of the examiner (Figure 3) is similar to that of the MSDOS acquisition utility:

- Case Number;
- Examiner [mandatory];
- Evidence Number [mandatory];
- Unique Description [mandatory];
- Current Time [mandatory];
- Notes.

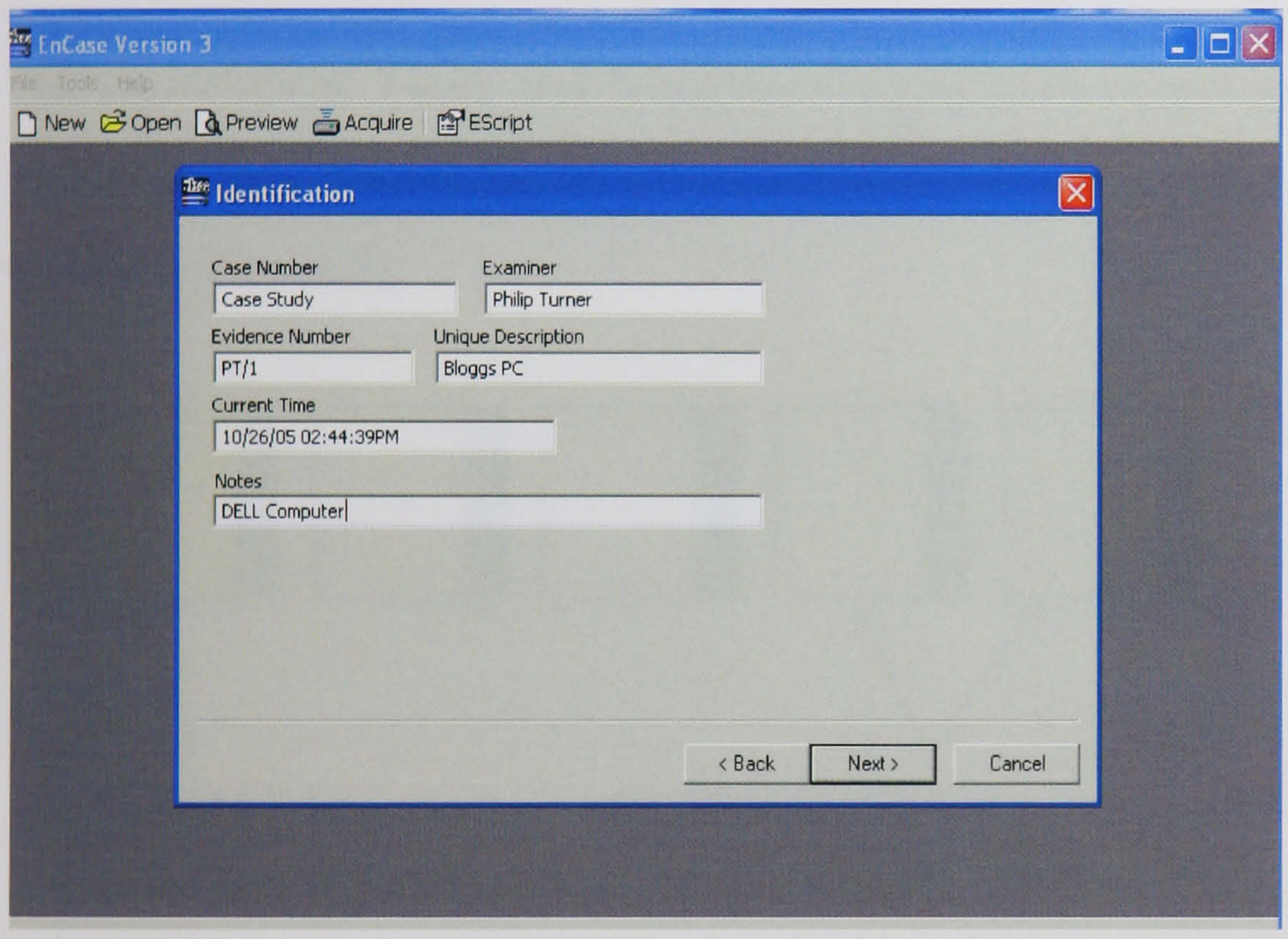

Figure 3 - Windows EnCase Acquisition screen 1

Other options that can be selected (Figure 4) when acquisition commences are:

- File Compression Level – None (Fastest, Largest), Good (Slower, Smaller), Best (Slowest, Smallest);
- Total Sectors to Acquire;
- Password – used to restrict access to the image file;
- Generate Image Hash – selects whether an MD5 hash is calculated and written to the footer of the image file at completion of the acquisition;
- File Segment Size – selects the output size of each image file. A size is usually selected that is convenient for the backup media capacity e.g. CD or DVD.

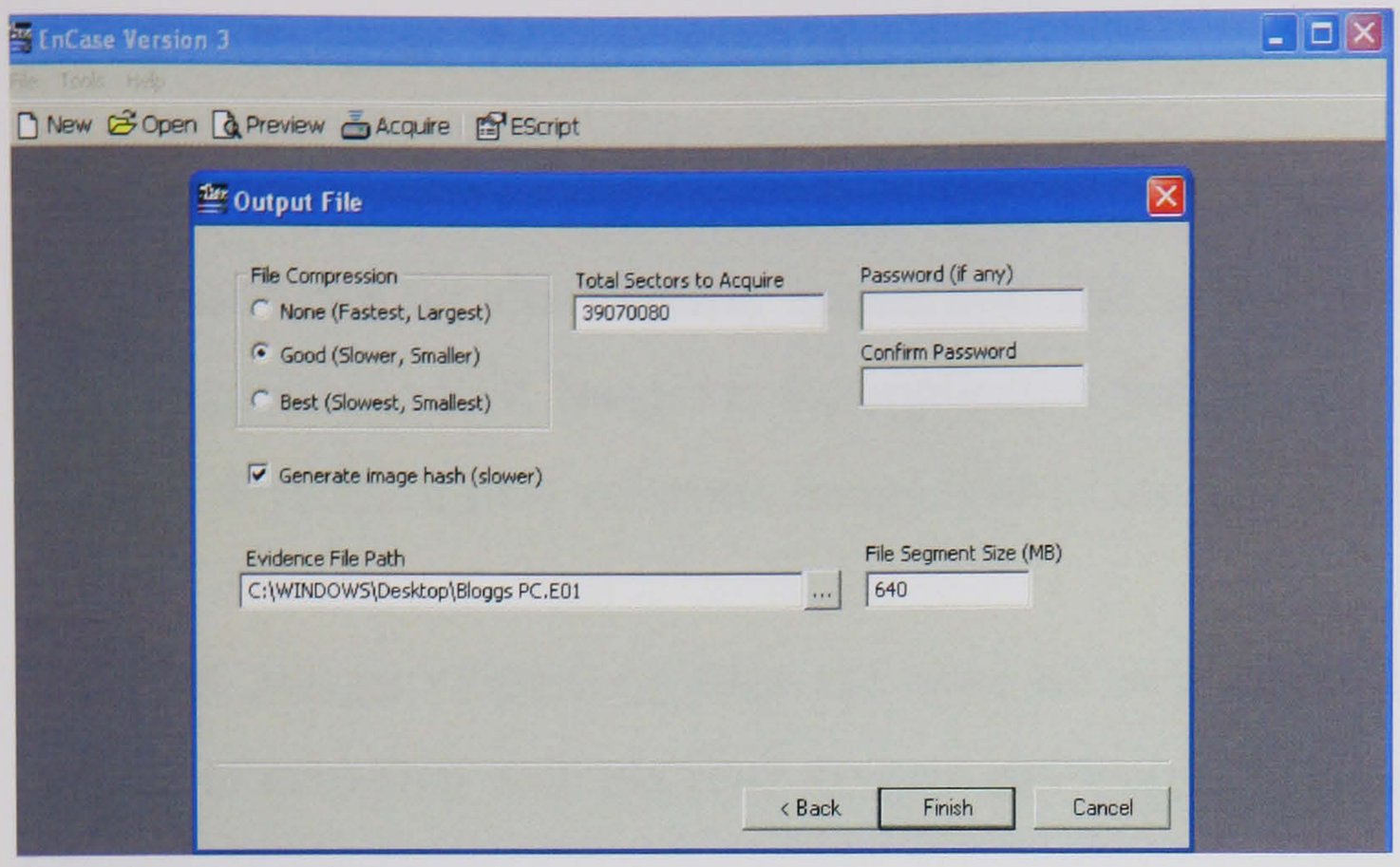

Figure 4 - Windows EnCase Acquisition screen 2

EnCase MSDOS, Linux and Windows imaging tools output an 'Evidence File' with the same basic format (*Bunting 2006*). This format is summarised in the following diagram (Figure 5).

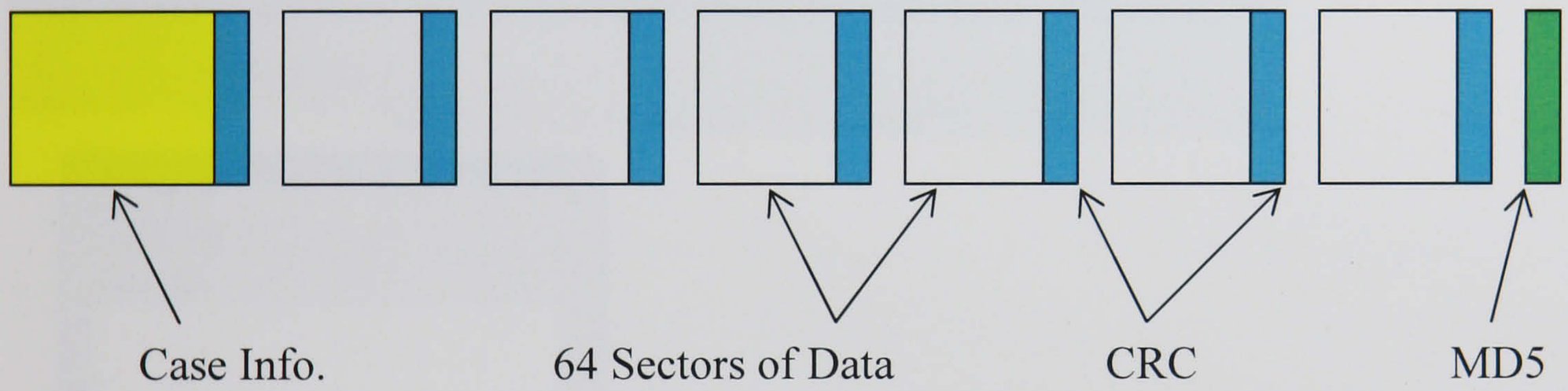

Figure 5 - EnCase Evidence File structure

The full specification of this format is proprietary and has not been released by Guidance Software. There are however a number of utilities such as Access Data FTK (http://www.accessdata.com) and Mount Image Pro (http://www.mountimage.com) that can open and interpret the Evidence File format.

The format records the location of bad blocks to a resolution of 64 sectors and fills the corresponding section of the image file with 'null' filled data. The problem with this approach is that a single bad sector encountered on the source device results in 64 sectors of null filled image file data.

2.2.1.3 Forensic Toolkit (FTK) Imaging

The Forensic Toolkit (FTK) and FTK Imager are two Windows applications produced by Access Data for the forensic acquisition and analysis of digital devices. As its name suggests the FTK Imager is the application that is used to acquire evidence from physical devices, logical volumes, image files or the contents of a folder.

The FTK Imager (Figure 6) does not have its own native image file format but is capable of acquiring both physical devices and logical volumes into image files of the following formats:

- Raw (dd) – plain bit stream image format;
- SMART (*ASR 2002*);
- E01 – EnCase compatible evidence file format.

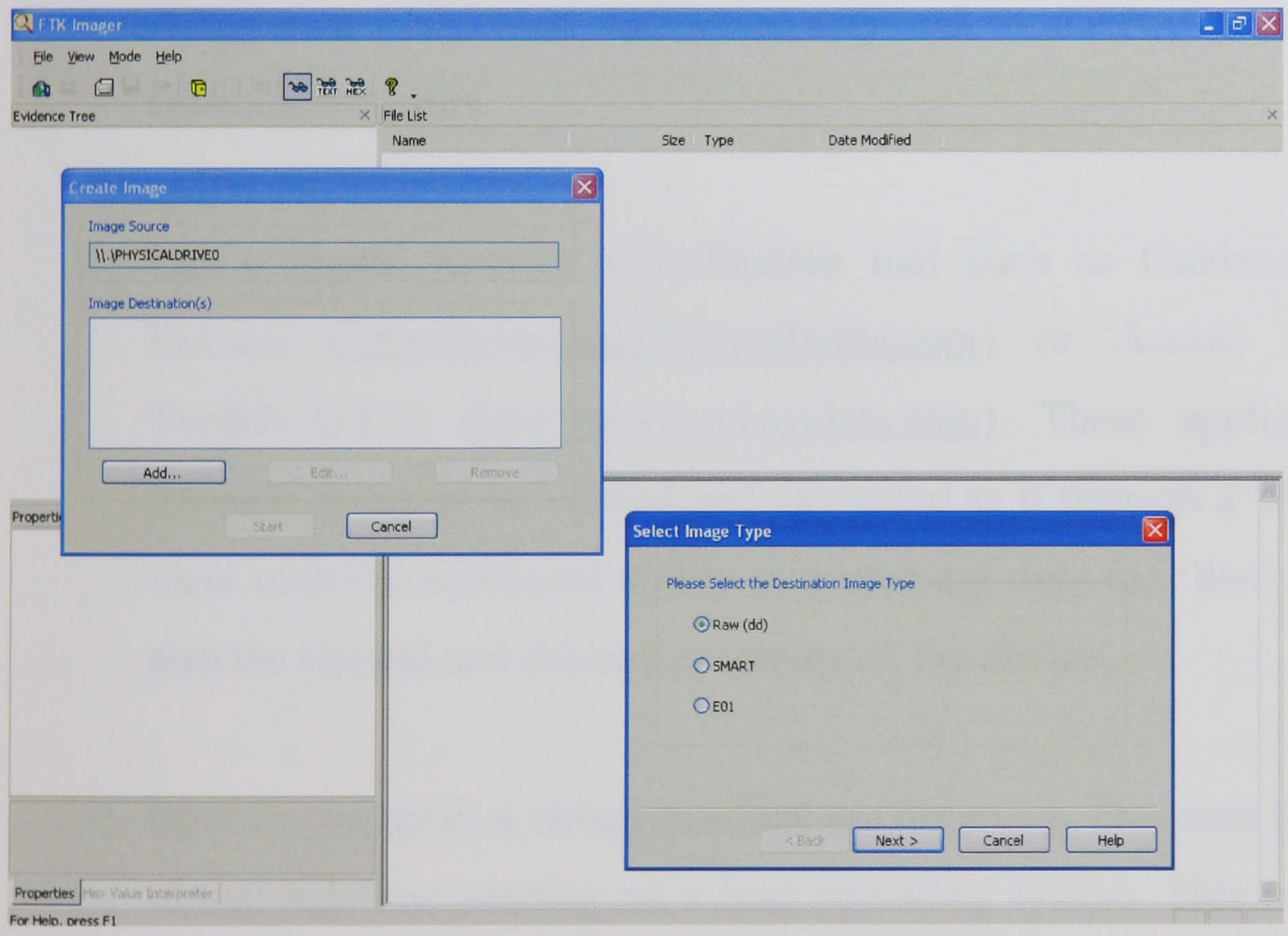

Figure 6 - FTK Imager Acquisition screen

2.2.1.4 Digital Imaging Summary

The previous sections discuss some of the commonly used digital imaging tools and highlight a number of inconsistencies in approach and therefore the evidence that would be captured. This makes it very difficult to conduct dual tool verification without fully understanding how each specific tool operates or handles media errors.

This situation is compounded because these basic details are often not published by the tool vendors and is left for the practitioner to discover empirically. This strengthens the case for standardisation in what is considered the cornerstone of the digital forensic process.

2.2.2 Digital Examination

Once a successful image has been acquired there are a number of approaches that can be adopted in order to examine the contents of the device image. These are:

1) Restore the image to a clean device to create a clone of the original and boot the device on either the same or similar hardware as the original. This used to be problematic as the hard disk geometry had to be identical to the original. However, since hard disks have increased in capacity (above 8GB) the addressing methods used to access them have become simpler (Logical Block Addressing (*Microsoft 2000*)) as opposed to CHS (Cylinder-Head-Sector) addressing modes.

2) Use a digital forensic investigation tool such as Guidance Software Inc. EnCase (http://www.guidancesoftware.com) or Access Data's Forensic Toolkit (FTK) (http://www.accessdata.com). These applications permit a forensic image to be viewed and navigated as if through a Windows Explorer view with the additional ability to access not only the 'live' file structure but also the unused and deleted structures on the device.

3) Boot the image in a virtual machine environment. This uses the technique of a virtual machine running on a host operating system. This has the advantage that a simulated environment can be created without the time and storage overhead of creating a clone. The problem with this approach is that it can occasionally be difficult to get the virtual machine hardware to simulate the original machine closely enough to create an operational virtual system.

Examination and investigation is a highly individual process. There is no prescribed manual on how to investigate and many good investigators 'follow their nose' and

develop their own processes and methods as experience increases. Typically. when using forensic analysis tools a number of basic processes will always be undertaken in a structured order regardless of the type of investigation being conducted. For example, it would be common to identify and recover deleted files or folders and also determine if the capacity of the logical partition actually filled the capacity of the device imaged. If any discrepancies were found then further investigation would need to be undertaken. Other basic tasks would also be undertaken; these may include performing a hash file analysis to eliminate known good files, identifying the version of operating system, the registered owner and any installed applications.

From then on, the type and objectives of the investigation have to be considered as these can determine if keyword searches, email examination, file carving (recovery of deleted file types), graphical file examination, document examination or identification and decryption of encrypted material etc., is performed. The list of potential tasks to undertake is numerous and constantly increasing as new applications, capabilities and features are constantly being developed.

A number of commonly used forensic examination and analysis tools are now briefly described to highlight the main features and differences between them.

2.2.2.1 Guidance Software Inc - EnCase

At present EnCase is probably the predominant commercial forensic analysis tool used by law enforcement. It provides an easy-to-use Windows based application for the examination of digital evidence. Its advantage over other tools is its ability to interpret a wide range of filing system formats; these include FAT (12,16,32), NTFS, EXT2, EXT3, HFS.

EnCase only supports EnCase evidence file format (E01) and raw image file formats.

The interface provided is a Windows Explorer style system. It allows the examiner to navigate the live and recovered files and view file contents in text, hexadecimal or gallery (picture) modes (Figure 7). File slack (the part of a file between the logical and physical end of the file) is easily identifiable in red.

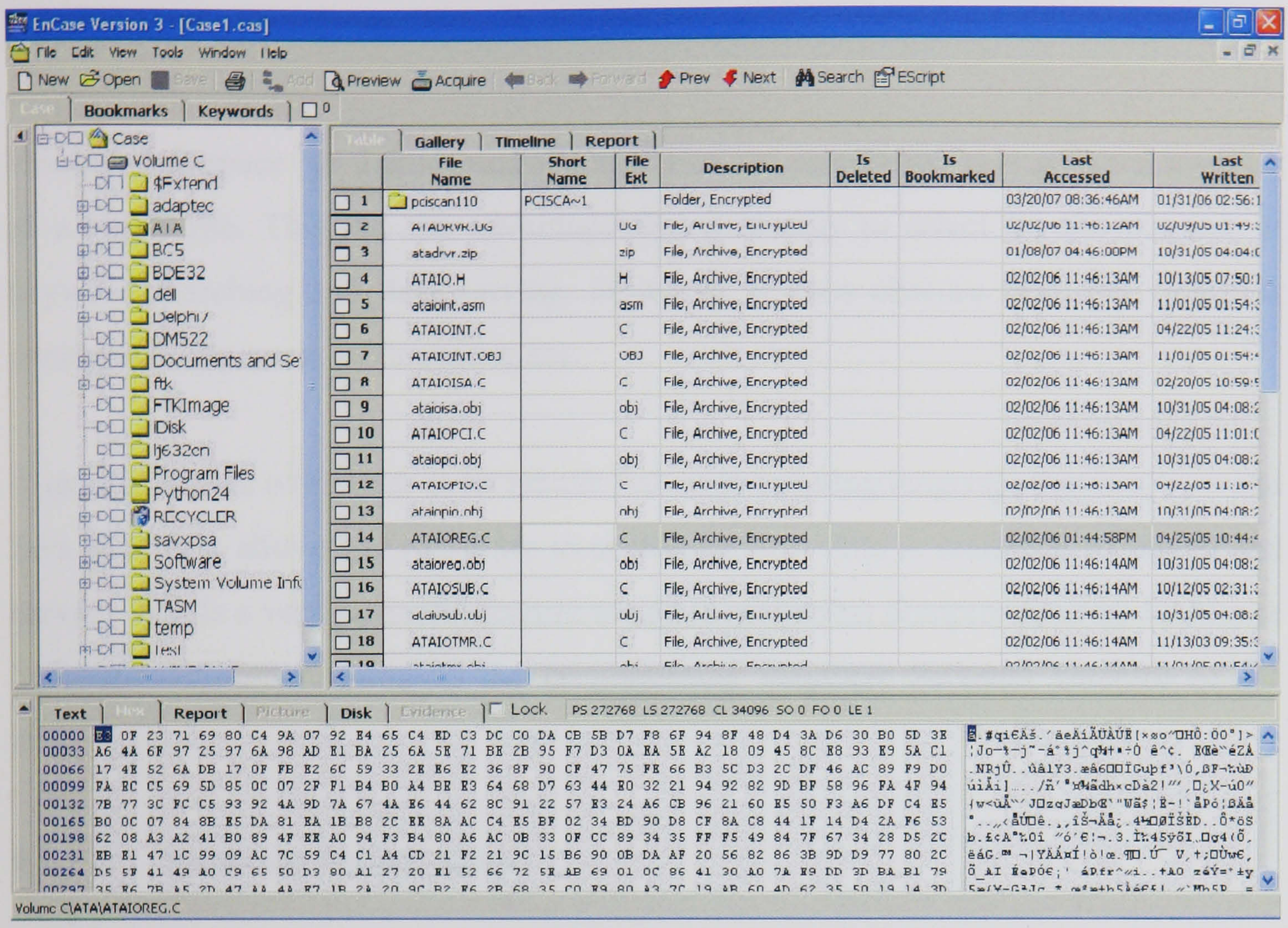

Figure 7 - EnCase analysis tool - screenshot

EnCase allows the examiner to search the evidence files for keywords (either case sensitive and / or Unicode) or GREP search patterns. It does this by searching the entire contents or selected files in a sequential manner. Therefore some searches can take a very long time to complete depending upon the volume of information selected.

EnCase allows a 'hash file analysis' to be performed. This is the identification of known and unknown files by comparing the case contents with that of a hash set comprised of certified known files. EnCase only supports hash sets based on the MD5 (*IETF 1992*) message digest algorithm.

A library of known hashes is available from the National Software Reference Library (NSRL) (http://www.nsrl.nist.gov) which is maintained by the National Institute of Standards and Technology (NIST). The NSRL database is a compilation of over 28 million unique file hashes containing MD5, SHA-1 and CRC signatures. Other cryptographic hash algorithms are planned to be supported in the future. Due to the demonstrated weaknesses of the MD5 algorithm (*Wang & Yu 2005*), there is a

progressive move towards SHA variants in preference to MD5 by the digital forensic community.

Unused disk space (or unallocated space in EnCase terminology) is grouped together as a single file. This has the advantage that it is easy to select for the purpose of keyword searching and data carving. However, it does obscure from the examiner contiguous sequences of unused space.

A unique feature of EnCase is its inbuilt EnScript scripting language that is a C++ and Java hybrid. It allows the examiner to customise how data is searched, extracted and carved. This is a very powerful feature which is somewhat compromised by Guidance Software constantly changing the programming language between versions of the tool.

2.2.2.2 Forensic Toolkit Suite

The Forensic Toolkit (FTK) suite of tools is produced by Access Data. It comprises a number of applications:

- Forensic Toolkit (FTK) - Main Forensic examination and investigation tool (Figure 8). This also includes the FTK Imager described in section 2.2.1.3;

- Password Recovery Toolkit (PRTK) - Utility for the cracking of encryption and other security mechanisms of over 50 applications;

- Registry Viewer - A utility for the examination of Microsoft Windows registry files;

- Distributed Network Attack (DNA) – A utility for distributed password cracking that utilises processor idle time to systematically search the key space of encrypted files.

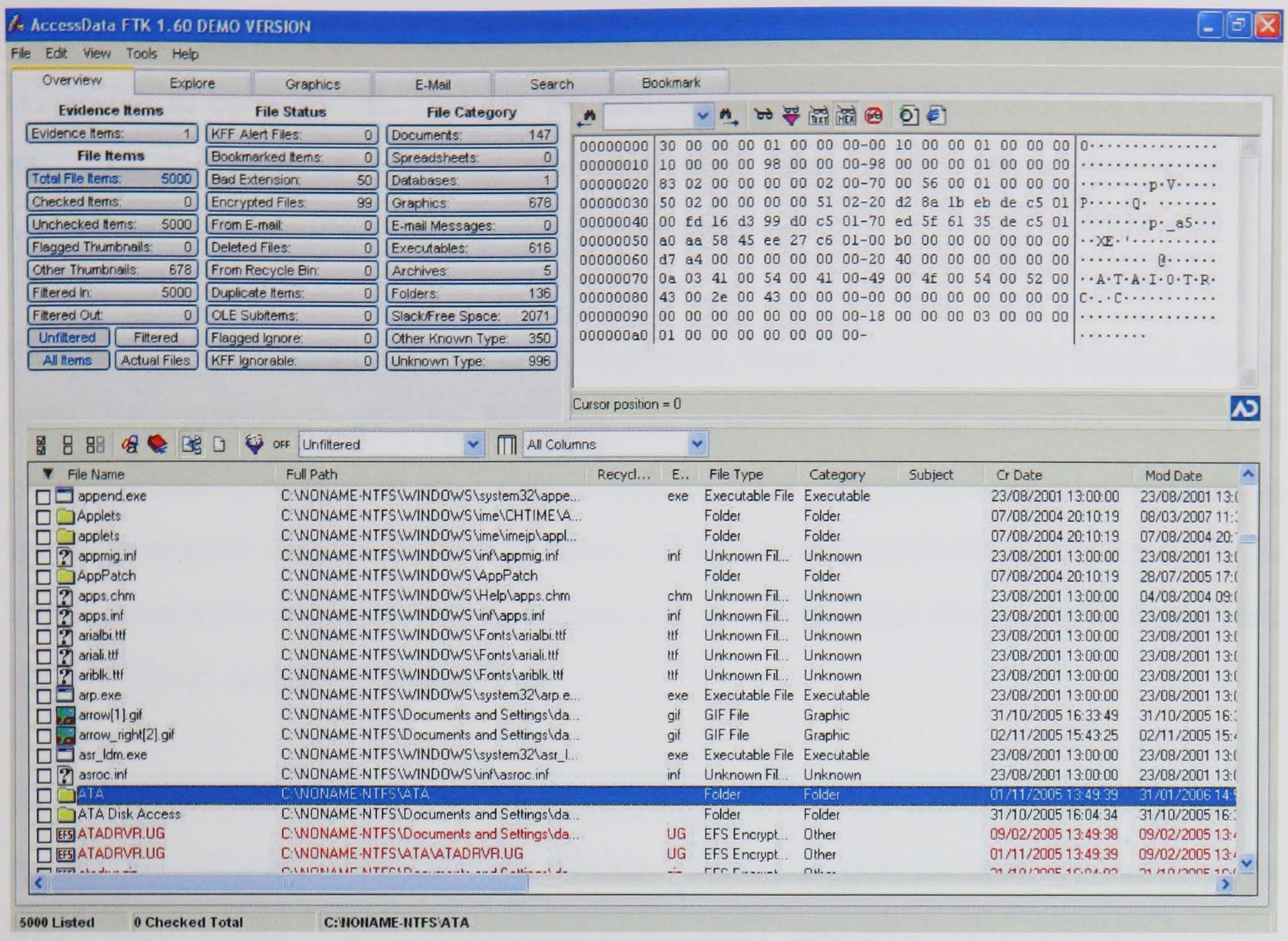

Figure 8 - FTK analysis tool - screenshot

FTK also allows the examination of the contents of an image file in a Windows Explorer style view. FTK supports a number of image file formats e.g. raw, E01, I01 to name but a few. However FTK's evidence examination paradigm is different to that of many other tools. When an image file is opened with this toolset it is first indexed to build a dictionary of all words identified within the image file and also entropy tested to determine the types of all files within the image. This process can take a while to complete (depending on capacity of image file loaded) but once performed allows the examiner to readily view how many files are in the following categories: Documents, Spreadsheets, Databases, Graphics, E-mail Messages, Executables, Archives, Folders, Slack/Free Space, Other Known Type, Unknown Type. In addition to this the File Status list is displayed that tabulates the following categories: KFF (Known File Filter hash set) Alert Files, Bookmarked Items, Bad Extension, Encrypted Files, From E-mail, Deleted Files, From Recycle Bin, Duplicate Items, OLE Subitems, Flagged Ignore and KFF Ignorable.

A keyword search within FTK is a very quick operation to perform, as it simply has to search the dictionary index and allows keywords to be combined within a user-

defined proximity. The index also allows a dictionary of all words found within an image file to be exported. This can be readily imported into the sister tool PRTK for password cracking of protected and encrypted files. This type of file is easily identifiable and exported from a case for this purpose.

FTK also performs a number of other functions when an image file is loaded. These include the automatic opening, indexing and reference of email container files. This permits email messages to be browsed and searched without having to go searching for the container files in the first instance. FTK also automatically opens and indexes compressed archive and storage file types, e.g. .CAB (cabinet files), .ZIP (compressed archive files), and .ISO (image files).

FTK supports MD5 (*IETF 1992*) and SHA-1 (*IETF 2001*) integrity hash algorithms for the identification of known files. Within FTK the support for hash sets is known as a KFF (Known File Filter). Importation of the NIST hash sets is also supported.

2.3 Problems with current methods

The scenario described in the previous section, demonstrates the basic process that would be undertaken from the search and seizure through to the imaging, examination and analysis as part of a digital forensic investigation. This basic scenario is useful to highlight a number of issues and limitations of the toolsets and methodology employed.

The following observations can be made:

1) The whole process is very reliant on the diligence of the investigator to keep sufficiently detailed notes of actions and processes performed through all phases of the investigation: *Preservation, Collection, Examination, Analysis* through to *Presentation* - see section 2.1.2. This aspect is also common to other forms of forensic science and not specific to digital forensics.

2) The reliability of the evidence that is obtained is also reliant on the investigator being adequately trained and competent to acquire the evidence without compromising the original.

3) Repeatability of the information obtained and processes undertaken throughout the examination are difficult to assure, as most tools do not log the actions performed on the evidence. Without this detailed logging it is almost impossible for an independent examiner to identify where, why or how an aspect of the investigation was performed.

4) When a traditional crime is investigated the methods and processes used should be proportionate to the severity of the offence (*TSO 2000*). In a digital investigation the approach and method adopted is rarely proportional to the severity of the crime being investigated; the imaging process does not take into account the type or objectives of the investigation. For example, when an image of a digital device is created the whole of the device is captured regardless of type of investigation, as this is seen as the only way to maintain the integrity of the information acquired.

5) The storage capacity of digital media is increasing at a phenomenal rate. This increases the time taken to acquire the evidence and therefore also increases the overhead required to process, search and backup.

6) There are no facilities for attaching associated material or metadata to a digital investigation. For example, textual notes, photographs, thoughts of the investigator, reasons and justification of taking a particular action. The metadata that is stored with the image is typically limited to name, place, date and time and an integrity checksum. This is often in a format that is proprietary and fixed by the facilities offered by the tool vendor. This also means that the formats are used to lock-in an investigator to one tool and restrict the use of multiple toolsets over the same evidence, thus hindering dual-tool verification.

7) Recording 'standard' practice and the errors or mistakes made during the course of an investigation is vital in order to provide the metrics for reviewing the

effectiveness and hence improving how investigations are conducted in the future. The types of error can broadly be divided into three categories:

a. Process – the failure to apply the correct processes and procedures that adequately take into account the circumstances of the investigation. This presumes the investigator knows what the correct process and procedures are and how to apply them;

b. Tool / Application Error handling – the tools that are used to acquire evidence do not correctly handle physical device errors or examination tools fail to interpret filing system or file formats correctly. For example, when an imager is unable to read the original source device the best that is usually done is to record the error location and pad the image file with null values;

c. Human error – mistakes are often made as investigators may be fallible, there may be mitigating circumstances or reasons for the mistake. This can result in the wrong processes, procedures or the application of the wrong tools and techniques that may then give unreliable results or outcomes.

The lessons learned from noting 'standard' practice and recording errors can be used to train new investigators and improve process, procedures and tools in the future if adequate facilities are provided to record, audit and review how investigations are undertaken. In a digital investigation the diligence of the investigator is relied upon to record the sequence of actions taken on the digital evidence.

8) The technological rate of change is very quick and as a result the forensic tools also have to be developed rapidly. This often compromises the amount of testing that is performed on a given tool before it is released for use. The reliance is often placed on the investigator to test the tools before use. More recently organizations like NIST have conducted a basic Computer Forensic Tool Testing programme http://www.cftt.nist.gov. However, these can only test a small subset of the core functionality provided by the most complex tools.

9) With the exception of the raw file format that is supported by all forensic tools but contains no provision for integrity or metadata, most formats are proprietary to a specific vendor. This means that evidence interchange between vendors' tools often requires extensive conversion techniques (usually back to a raw format) thus incurring what can be an excessive overhead in terms of processing time, integrity checking and storage media for no evidential gain and hinders dual-tool verification.

10) Digital forensics evidence capture is usually a static approach in a very controlled environment. This process totally disregards the requirement to acquire material in a forensically sound manner from a dynamic 'live' environment. Often the live environment is handled using standard system administration tools and utilities at a time when the forensic investigator may already be stressed due to the circumstances of the environment and immediacy of the incident being examined. This can often lead to irrational and reflex actions, and may cause poor note taking and hence a lack of continuity and repeatability of evidence capture.

11) When digital evidence is processed, the entire image file has to be available and loaded into the examination tool. There are currently no examination tools that allow multiple investigators to concurrently conduct an examination of a single instance of the evidence without the overhead of creating additional copies. Similarly there are no tools that allow multiple processors in a distributed environment to process evidence. The closest system to this is the Access Data's DNA tool that is used to brute-force a limited number of cryptographic applications.

12) Current forensic toolsets provide no features to track evidence continuity and log when an application is used to access the evidence.

13) Current forensic toolsets provide few facilities to trace what processes or procedures have accessed the evidence, what functions were performed or what results were obtained. This functionality is usually limited to the addition of a bookmarked item with a short textual comment.

14) All tools rely on the whole of the captured evidence being made available to the analysis tool before examination can commence.

15) No current tools allow the objectives or type of investigation to be taken into account when collecting or acquiring the evidence. The investigator does this when the examination of the evidence is undertaken.

2.4 Imaging and Analysis Tool Requirements

The problems identified with current methods, and the observations made in the previous section highlight a number of issues and shortcomings with current processes, tools and difficulties encountered due to technological advancement. It also shows that even though the technology and the capability of IT systems have been evolving, the digital forensic process has been relatively static, relying on physical measures to fulfill the needs of different jurisdictional evidential processes.

However, an underlying set of requirements have been identified that should be incorporated into current digital forensic imaging and analysis toolsets that complement and enhance the physical measures employed. They are:

1) To provide a mechanism to verify that the data presented at court is identical to that acquired from the original media, this may allow detection of errors or corruption or provide a mechanism to prevent deliberate unauthorised modification;

2) To record an audit trail of when, where, what or who made:
    a. Assumptions;
    b. Thought processes;
    c. Applications;
    d. Actions;
    e. Functions;

    used to Preserve, Collect, and Examine the digital evidence. This should be in sufficient detail to permit repeatable and independent verification of examination results and findings. It can therefore be used to highlight

problems and errors in processes, methods, tools and judgment thereby permitting it to be used for training purposes and aid certification;

3) The mechanisms should be able to be tailored and proportionate to the investigation being undertaken;

4) To provide a mechanism to store evidence as efficiently as possible to improve processing, searching and backup;

5) To provide a means of storing associated information and metadata;

6) The storage format should be open and well defined;

7) The storage format should accommodate multiple investigators and systems to simultaneously access the evidential information;

8) The storage format should be able to accommodate information obtained from a disparate range of source devices within static and dynamic environments.

The requirements listed above can be correlated with the problems discussed in section 2.3. There is not a direct singular mapping as some problems may result in multiple requirements, Table 1 summarises this correlation.

Identified Problems – summary descriptions described in section 2.3	**Requirements – summary descriptions described in section 2.4**
1) Reliance on diligence of investigator to keep notes.	**1)** Record provenance.
1) Reliance on diligence of investigator to keep notes. **12)** Current tools do not record access to evidence. **13)** Current tools do not record process or function applied to evidence.	**2)** Record continuity.
8) Lack of tool testing. **9)** Proprietary evidence format.	**1)** Maintain integrity.
1) Reliance on diligence of investigator to keep notes. **3)** Lack of logging. **6)** Inability to store associated meta-data. **7)** Not noting errors or mistakes. **12)** Current tools do not record access to evidence. **13)** Current tools do not record process or function applied to evidence.	**2)** & **5)** Record audit trail of process / assumptions / through processes / application used / actions / functions / errors.
3) Lack of logging. **7)** Not noting errors or mistakes. **8)** Lack of tool testing.	**1)** Reliability & **2)** Repeatability.
4) Not taking account of type of offence being investigated. **14)** Entire evidence capture before commencement of examination. **15)** Not taking account of objectives of investigation.	**3)** Proportionate.
2) Competency of investigator.	**2)** Training / Education aid.
5) Rapid increase in volume of information. **14)** Entire evidence capture before commencement of examination.	**4)** Efficiency – processing / searching / backup.
8) Lack of tool testing.	**2)** & **6)** Tool certification / testing.
11) Current tools only permit single investigator examination.	**7)** Multi-investigator support.
10) Traditional digital forensics techniques operate only on static data.	**8)** Support for disparate range of digital devices and environments.

Table 1 - Digital Forensics Issues & Requirements Correlation

2.5 Scope of Thesis

The author has many years experience working in the field of digital investigations and it is considered worthwhile re-examining the processes that are the cornerstone of digital forensic investigations. The aim of this study is to address the issues and problems identified in section 2.3 and propose a solution that is capable of fulfilling the requirements defined in section 2.4. The solution will be primarily directed towards the acquisition (imaging) and subsequent examination of computer based media, as this is the area of digital forensics that is currently most widely understood.

Specifically, if additional audit and continuity features can be supported then the absolute reliance on the diligence of the investigator can be reduced so that they are more able to focus upon the investigation rather than the procedural methodological record taking. This reduces the possibility that cases could be dismissed on procedural technicalities and is also an aspect that should be able to be implemented in modern systems without significant overheads. Additionally, the verification of results or the ability to determine if inconsistency is due to an analysis application error is just as important.

Also, the ability to provide a proportional approach to the capture of evidence is addressed, because with the ever increasing media storage capacities it may not be efficient or economical to always capture everything.

Current forensic tools operate in either a live (dynamic) environment or static environment. A framework that is able to accommodate both of these scenarios should enable common and more efficient analysis tools to be created as the type of functions that are required to be performed on evidence captured from both environments are identical; for example, keyword searches, dictionary creation, information extraction or timeline analysis.

Ideally an extensible solution should be able to accommodate current technology and cater for future developments in storage capacity and alternate digital devices as they are developed and become prevalent in society.

3 Digital Evidence Bags (DEB)

3.1 Traditional Evidence Storage – Bags, Tags and Seals

The tried and trusted method for containing and storing physical material that may be used evidentially is the evidence bag. Although there are no international standards for physical evidence bags, they all serve two purposes:

1. To securely contain the object so that it can be detected if it has been accessed or tampered with;
2. To provide a vehicle to record the provenance and continuity of its contents until such time as it may be used in a court or tribunal.

There are a number of common sizes of bag produced, but the actual size used is at the discretion of the seizing official and would be chosen depending upon the size of the article that it has to contain. The type of bag used would also be selected depending upon the type of material that it has to contain and is manufactured from either a transparent or opaque material. The reason for using an opaque container would be if the material seized should not be viewable until it was examined officially. For example, if potential legally professional privileged (LPP) documents were seized that may contain sensitive or restricted material, which may be relevant to other legal cases; this should only be viewed for the purpose of the particular case under investigation. Other evidential material such as a desktop computer may be stored in a transparent bag as there is no compromise of the material or chance of viewing sensitive information until the system is removed for examination.

The following picture (Figure 9) shows a typical transparent evidence bag with integrated tag. An evidence tag stores key information regarding who, where, when and a brief description of what evidence was seized. The tag is used by the investigating officer to record details about the material collected and should be completed at the time of seizure. The following information can be recorded:

- Exhibit identifier – the exact format of this may vary according to local convention. The following are commonly used:

- The seizing officers initials followed by a reference number e.g. PBT/1;
- The day, month, year and then the seizing officers initials and reference number e.g. 24112007/PBT/1;
- The laboratory case reference number followed by the seizing officers initials and reference number e.g. ABC/123/PBT/1;

- Brief description of item/contents;
- Where, when and by whom the item was seized;
- A signature of the seizing official;
- An area to record the details of where, when and who took custody of the item after the item was seized. This is used to provide continuity of the evidence from time of seizure until used as evidence, restored to the owner or destroyed.
- A tamper-proof seal and 'pseudo' unique number. The seal number format is specific to a manufacturer. However it is possible to have identical seal numbers although the chance of these being used in the same case is minimal.

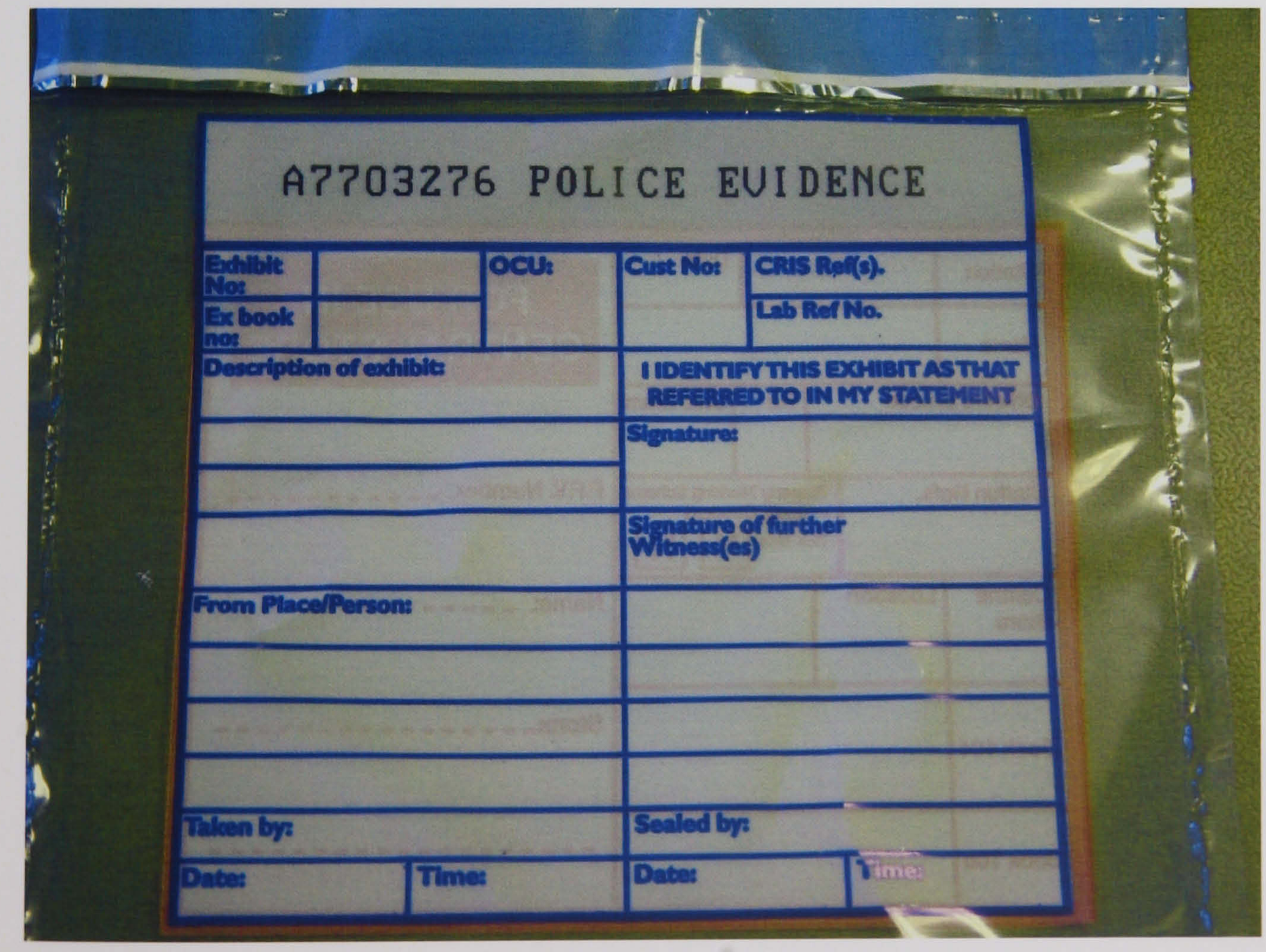

Figure 9 - Transparent evidence bag label

The following pictures show a tag (Figure 10) that would be used in conjunction with bags without an integrated tag and tamper-proof seal (Figure 11 and Figure 12).

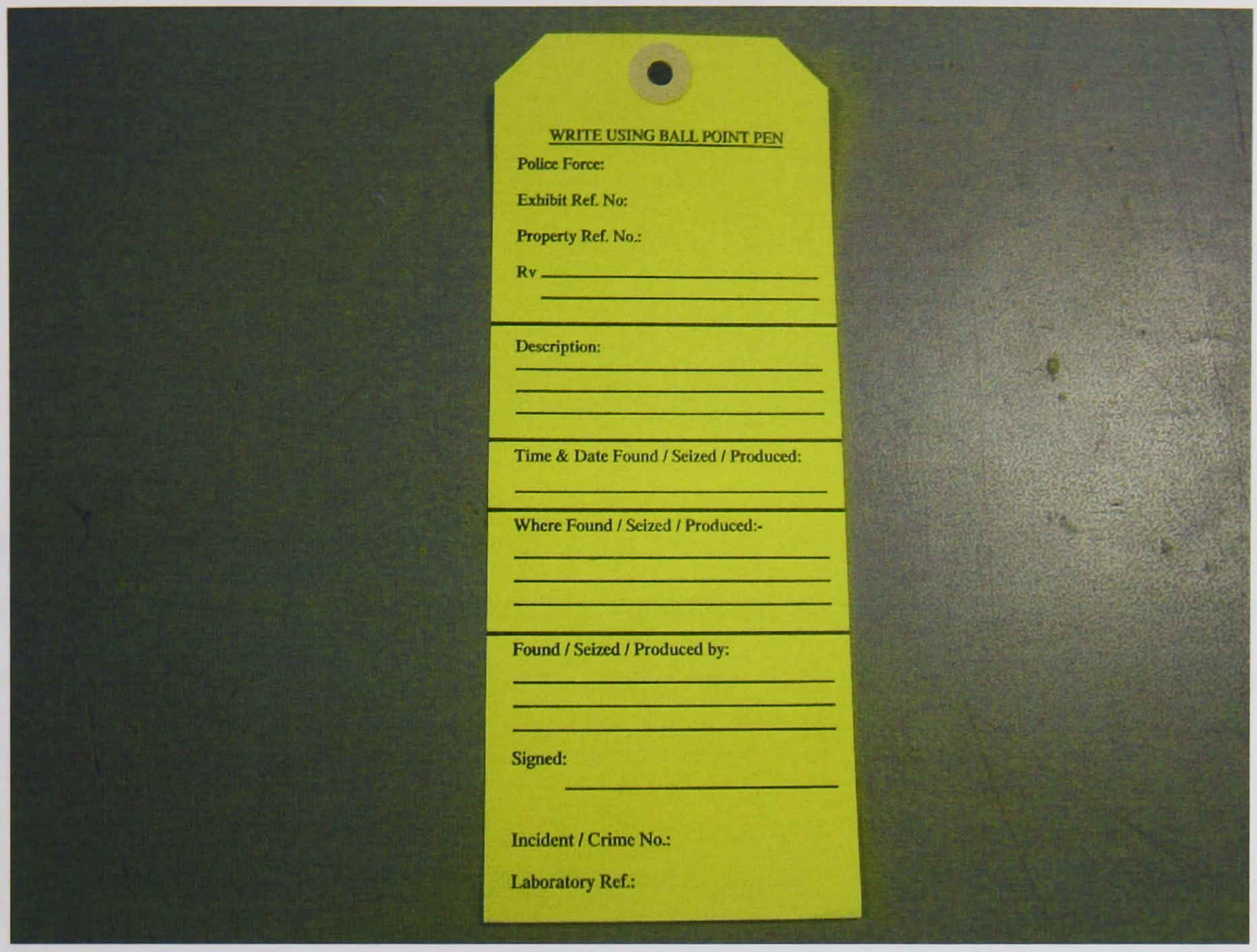

Figure 10 - Evidence tag

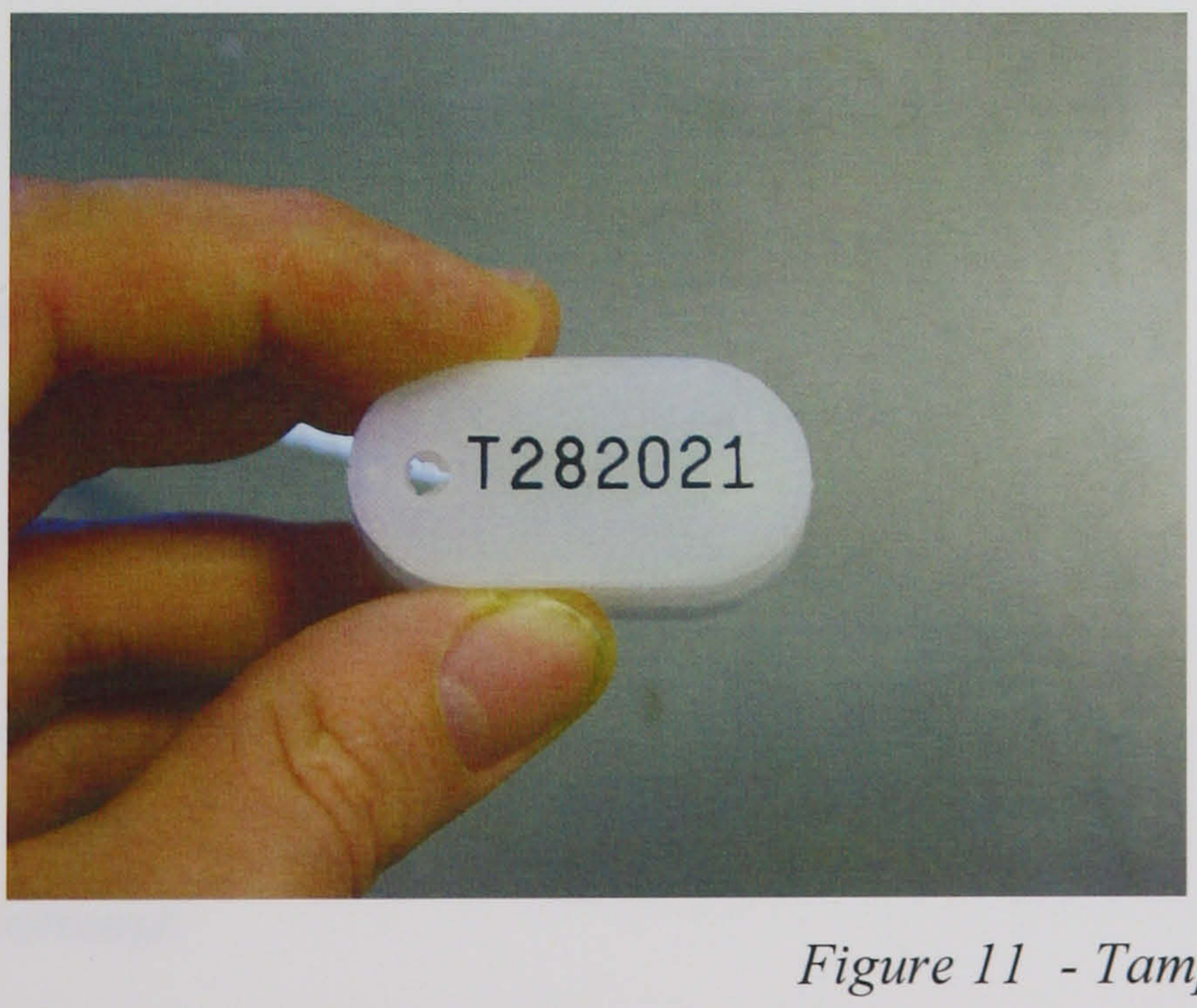

Figure 11 - Tamper-proof seal - 1

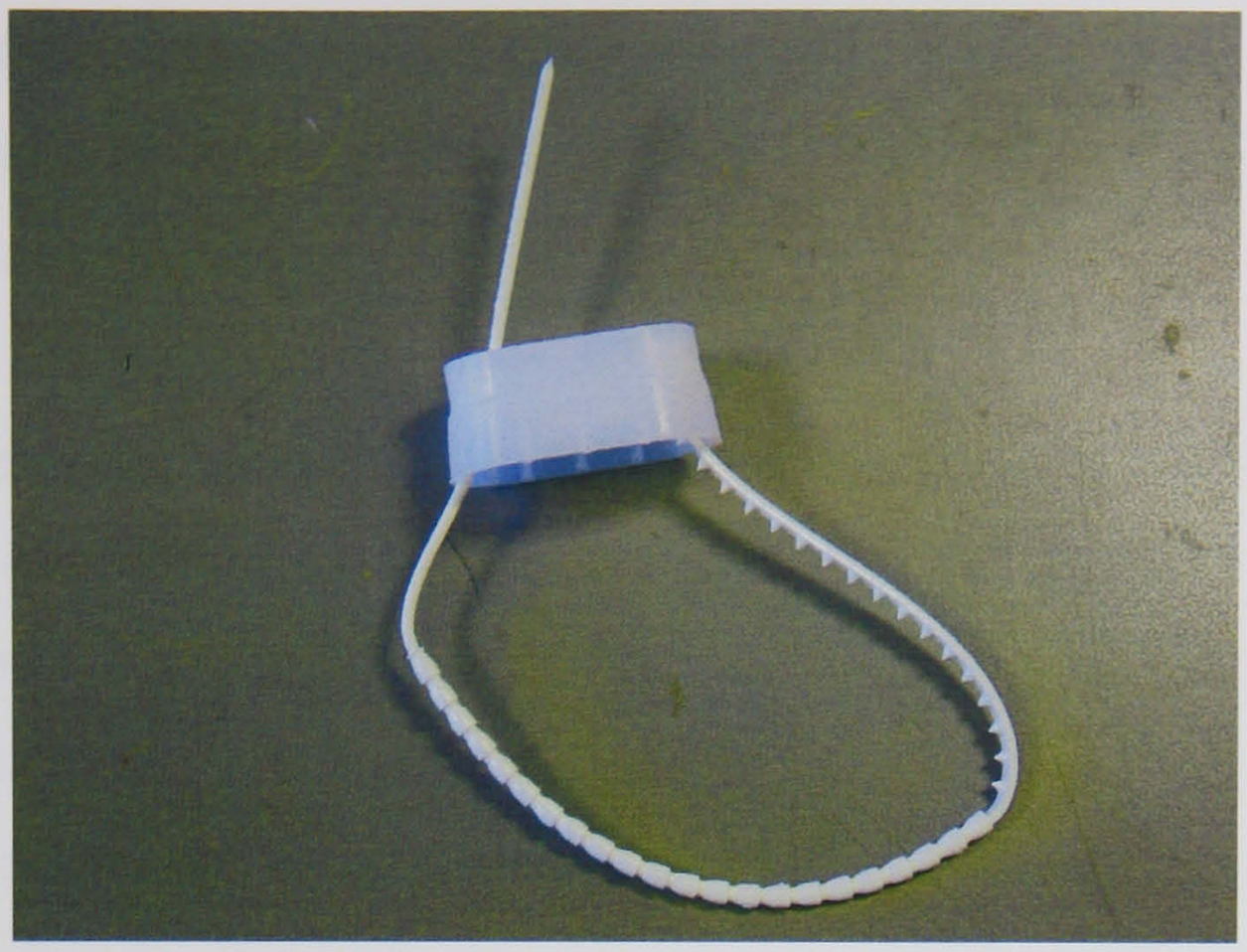

Figure 12 - Tamper-proof seal - 2

The concept of bags, tags and seals is universally accepted, and understood by all jurisdictions, law enforcement personnel, legal professionals and to a lesser extent the general public. All digital evidential materials rely upon the physical evidence handling and storage methods to restrict access to the physical storage media.

3.1.1 Characteristics of Bags, Tags and Seals

In summary the following characteristics of Bags, Tags and Seals used in traditional evidence storage and handling are:

- Bags are universal containers used to store many different types and quantities of physical evidence. They may be opaque or transparent depending upon the type of material to be stored;
- Tags provide a mechanism to record provenance and track continuity of the evidential material contained in the bag;
- Seals provide a mechanism to detect the unauthorised access to the evidential material contained in the bag.

3.2 DEB Concept

As discussed in the previous chapter the format of files used for the storage of digital evidence is not standardised, and usually only record limited provenance information. The Digital Evidence Bag (DEB) addresses these shortcomings of the digital formats currently in use by emulating the characteristics of the physical bags described in section 3.1.1. This is done because the judiciary and law enforcement understand and

accept the use of physical evidence bags, however within the digital world additional features can be implemented to further enhance evidence integrity assurance, record provenance and track continuity.

A DEB is a universal extensible container for the storage of digital information acquired from any digital source (*Turner 2005b*), the format of which can be manipulated to meet the requirements of the particular information that is to be stored. The format definition is extensible thereby allowing it to encompass new cryptographic and compression algorithms and protocols as developed in the future, whilst also providing the flexibility for some degree of backwards compatibility as the format develops.

The basic structure of a DEB is shown in Figure 13 and comprises of three components – Tag, Index and Bag. The terminology used to describe the main components was chosen to replicate that used for traditional evidence handling for ease of understanding and clarity of purpose. This shows a hierarchy of components of which the tag is used as the primary container recording Evidence Unit content descriptions, evidential integrity and continuity records. An Evidence Unit (EU) is a pair of components (Index and Bag) that contain any arbitrary evidential data in the bag and a tabular list of the bag contents in the index. This structure was chosen because of its flexibility, simplicity, comprehensibility and scalability.

A DEB and most other digital forensic storage formats provide evidential integrity assurance mechanisms that are stronger than the seal numbers used for physical evidence, usually in the form of a cryptographic digest or hash. However, these only allow for the detection of accidental or media corruption changes. The DEB concept could improve upon this by storing in one EU the necessary information required for evidence error correction or even repair.

A 'DEB aware' application is a forensic tool that is capable of parsing the tag to identify and access Evidence Units (EUs) applicable to the function of the utility, and maintains the evidence continuity audit trail by updating the tag appropriately. In order to be able to maintain the integrity, continuity and auditing facilities that the DEB framework provides, a 'DEB aware' application should implement a standard

API that controls the creation and access of the various DEB components. A standard API provides the additional advantage that forensic applications can be more easily tested and verified.

3.3 DEB Components

The following lists the DEB components that are created as part of the creation and capture process, hence for a single evidence capture three components are created (Figure 13):-

- Tag;
- Index nn – where 'nn' is a reference number/count;
- Bag nn – where 'nn' is a reference number/count.

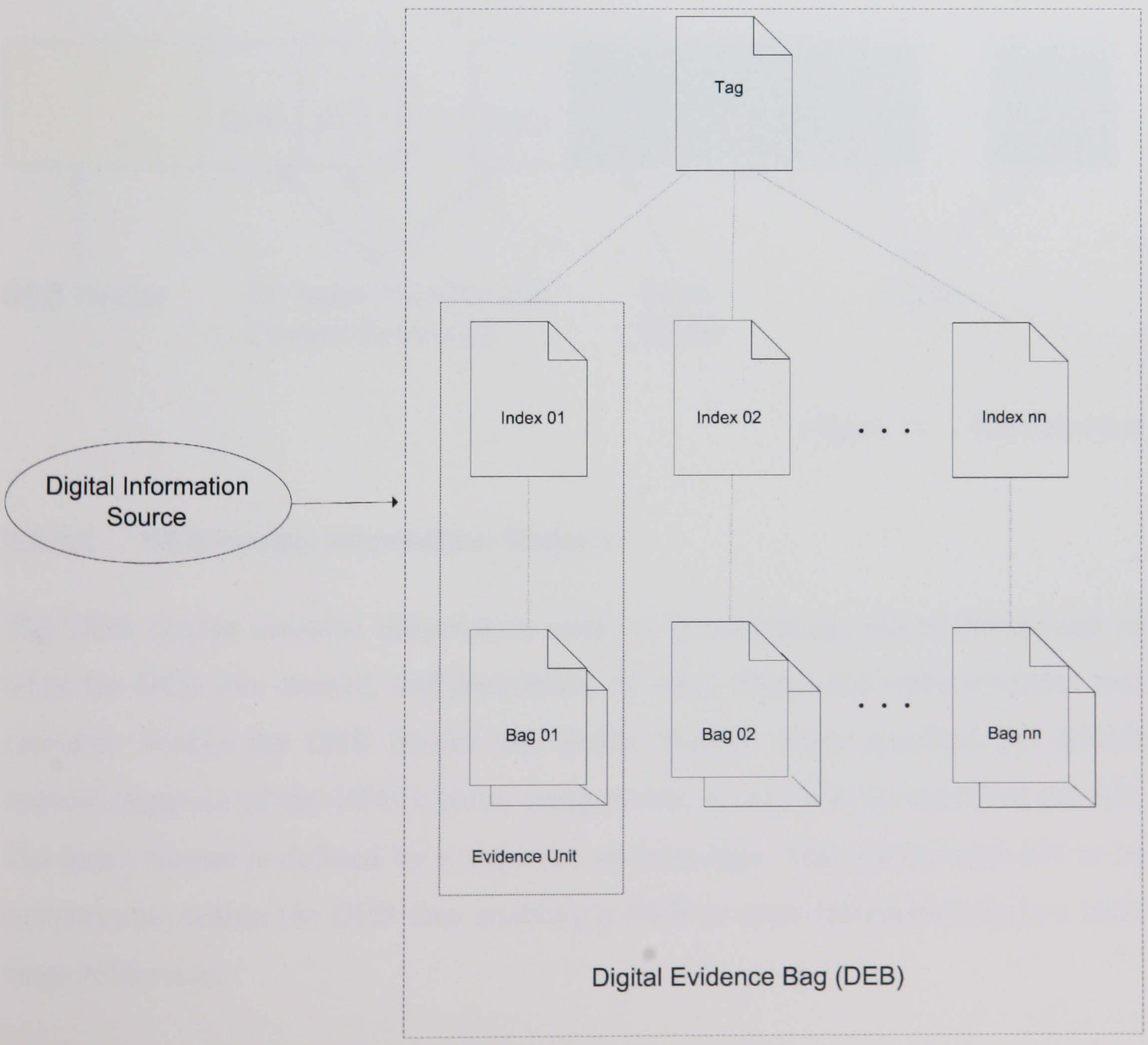

Figure 13 - Digital Evidence Bag basic structure

3.3.1 DEB Tag

The Tag component of a DEB comprises of four main information sections (Figure 14):

- DEB Header;
- Evidence Unit (EU) index number and content reference;
- DEB Footer;
- Tag Continuity Blocks (TCBs).

The Tag is a dynamic component of a DEB. When a DEB is created the Header to Footer sections are recorded sequentially, with the Footer being written on termination of the DEB creation/evidence capture application. All subsequent applications that access any contents of the DEB are recorded in TCBs.

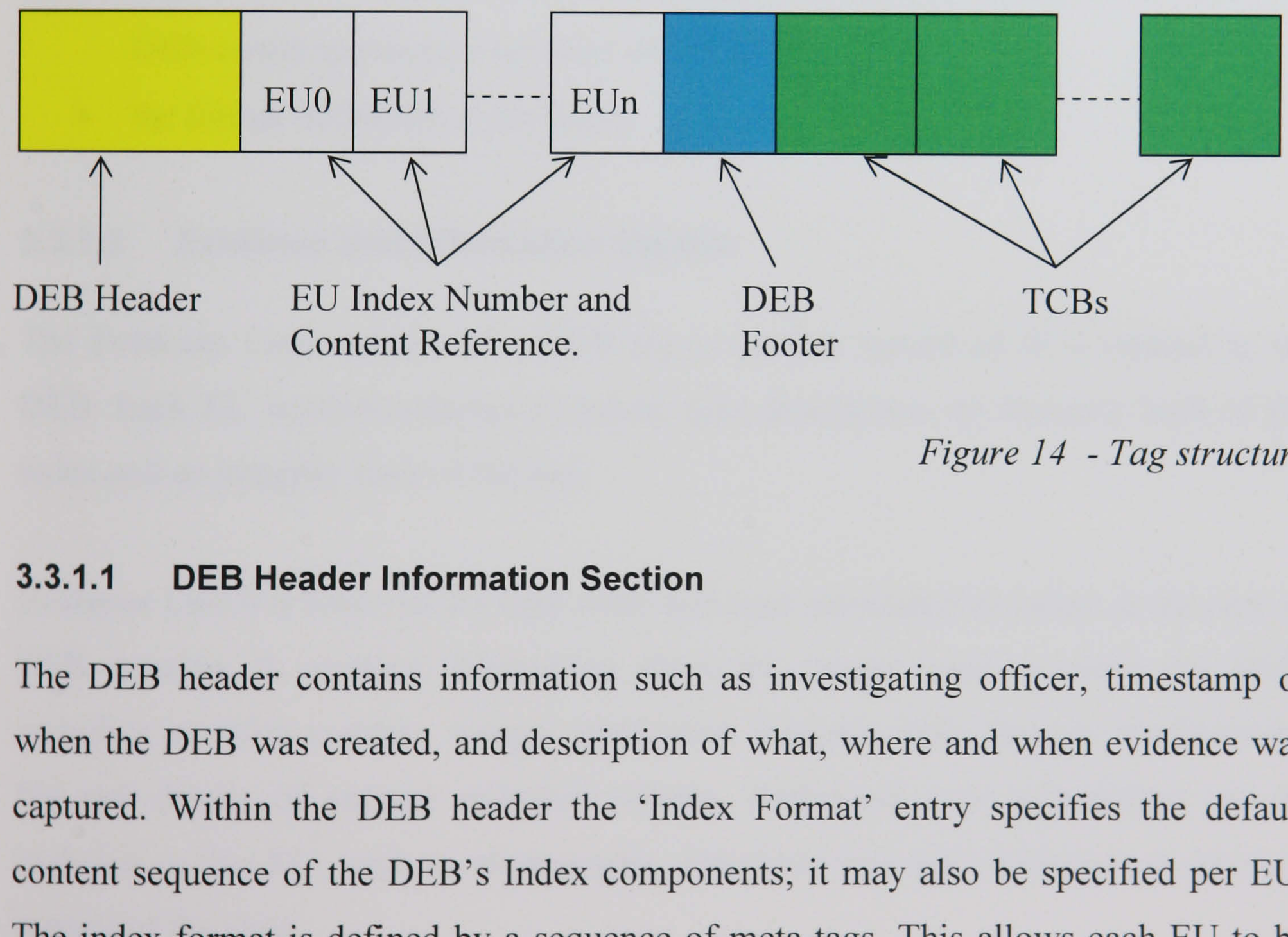

Figure 14 - Tag structure

3.3.1.1 DEB Header Information Section

The DEB header contains information such as investigating officer, timestamp of when the DEB was created, and description of what, where and when evidence was captured. Within the DEB header the 'Index Format' entry specifies the default content sequence of the DEB's Index components; it may also be specified per EU. The index format is defined by a sequence of meta-tags. This allows each EU to be customisable within the DEB thus enabling a DEB to store information from a wide range of devices.

The information recorded in the DEB Header is analogous to that recorded on an evidence tag when a physical item of evidence is seized, see Figure 9 and Figure 10.

The tag contains the following information:

- DEB unique reference identifier;
- details of the evidence contained in the DEB in a machine parseable form;
- the name and organisation of person capturing the information;
- the date and time the capture process started;
- a list of Evidence Units (EUs) contained in the DEB. An EU is the name given to an Index and its corresponding Bag;
- a hash of the captured information contained in the DEB;
- tag seal number comprised of a signature/hash of the tag to date, this is equivalent to the traditional seal number;
- Tag Continuity Blocks (TCBs) containing continuity information of when any DEB aware application accesses the DEB;
- the format definition of the Index.

3.3.1.2 Evidence Unit Information Section

The Evidence Units section of a DEB tag is used to record all EUs created in the DEB. Each EU section contains a content type description, an integrity hash of the index and an integrity hash of the bag.

Evidence Unit 0 is reserved for case notes and case associated metadata at the time of DEB creation. It contains information about the imager used to create the DEB, including revision number, imager application integrity hash, imager configuration file and details of capture selection criteria. Additional case information can be included in this EU, such as photographs, free-form text and references to physical aspects of the case.

The content type of other Evidence Units is arbitrary and is determined by the examiner based upon the case requirements and / or configuration of the imager or acquisition tool and device being acquired.

3.3.1.3 DEB Footer Information Section

The DEB Footer section is used to record the number of EUs within the DEB. The DEB is then sealed with a tag signature / hash written at the end. This is analogous to the signature of the seizing officer on a physical evidence tag. Subsequent access to the DEB contents result in a Tag Continuity Block (TCB) being appended to the tag.

3.3.1.4 Tag Continuity Block Information Section

DEB aware applications update the tag with a Tag Continuity Block (TCB) to reflect the history of operations performed on a particular EU or DEB as a whole. TCBs are used to record the open and close operations performed on a particular EU or DEB.

The information recorded in a TCB includes:

- the date and time a DEB aware application was used to access an EU or DEB;
- an application signature / hash;
- an application description and version number details;
- an application category description;
- an updated tag signature / hash. This is a newly generated / calculated value covering the revised tag data.

The TCB information section of the tag contains an expanding set of TCB details. Tag continuity sections used for physical evidential items (see Figure 9 and Figure 10) typically only record the signature and timestamp of the person taking custody of the evidence. In this respect the DEB TCB records more comprehensive information regarding the access, function performed and handling of the evidence.

Ideally the actions performed on a DEB and the transactions recorded within the TCB will be performed reliably. To achieve this, the DEB tag is required to have the following properties:

- Atomicity - refers to the DEB API being able to guarantee that either all of the tasks of a transaction recorded in a TCB are performed or none of them are;

- Consistency - refers to the DEB tag being in a legal state when the transaction begins and when it ends. This involves performing integrity checking of the DEB tag and EU components.
- Isolation - refers to the ability of the DEB application to make operations in a transaction appear isolated from all other operations. This would be particularly difficult to achieve in a multi-processor or distributed environment.
- Durability - refers to the guarantee that every transaction is recorded and that once the user has been notified of success, the transaction will persist, and cannot be undone.

ACID properties are typically defined for distributed systems and transactional database manipulation (*Coulouris et.al 2000*). An API that encompasses these properties also fulfils some of the forensic application requirements defined in section 2.4, Table 2 shows the correlation between ACID properties and the requirements that they assist in fulfilling.

ACID Property	**Requirement**
Atomicity	Maintain Integrity
Consistency	Maintain Integrity Repeatability/ Reliability Record Audit Trail
Isolation	Maintain Integrity Multi-investigator Support
Durability	Maintain Integrity

Table 2 - ACID Properties & Requirements Correlation

The ACID properties are highly desirable, the degree to which these are achieved in any particular DEB implementation affect the environment (single or multi-user and single or distributed processing system) in which it is deployed and thus the flexibility and usability available to the digital forensic investigator. The TCB provides the basic mechanisms required of a digital evidence handling and storage system to maintain the integrity, record an audit trail and thus a method that should be repeatable and

reliable. The specific implementation details required to implement the ACID properties within the DEB context are not considered further in this thesis and is an area which should be undertaken in future DEB API development.

3.3.2 DEB Index

The index details the contents of the corresponding bag. The index may contain details such as a list of file names, folder paths, and timestamp information relating to the contents of the digital information in the bag. Alternatively it may contain details of the physical device, for example the make, model and serial number of the device seized or device from which the digital evidence was obtained. The exact contents and format of this component varies depending upon the content type of the EU. Its format is specified in the Index Format definition in the tag and is comprised of a series of Meta-Tag labels that define its contents.

3.3.3 DEB Bag

The bag component contains the actual evidence captured. The contents of this bag may be either raw binary information (e.g. from raw device capture), files (e.g from logical volume acquisition), structured text (e.g. from network packet capture) or categorized files (e.g. one bag containing all text files, another containing all MS word docs, another containing all JPEG graphics files etc.). Additionally, a bag could contain another DEB.

3.4 DEB Characteristics

3.4.1 Provenance

One of the most important requirements for traditional evidence handling is to be able to record the provenance (section 3.1.1) and track continuity of evidence from time of seizure until used as evidence in court, and for a period of time after the case has been heard, for some offences this can be up to 30 years under UK law. Although this is one of the prime requirements no formal definitions as to how this should be applied to digital evidence exist. To clarify this, the author has given consideration into the definition of a number of attributes which together constitute evidence reliability

(*Turner 2005*) (*Turner 2007a*) - a copy of which may be found in Appendix A. These attributes of 'good' provenance are defined as:

- Unique - the provenance of a piece of evidence shall resolve to a single instance of that evidence. There may well be multiple copies of an identical file on a system but the provenance of each shall be unique, therefore it is possible to have multiple identifiers for a single instance or entity.

- Unambiguous - the provenance of a piece of evidence shall not be open to misinterpretation.

- Concise - the provenance of a piece of evidence shall be succinct and clear. This may not always be possible if using a logical location that may have been defined by the user of a system.

- Repeatable - the provenance of a piece of evidence shall be simple to replicate thus confirming the evidence provenance. This should be true independent of the forensic tools used during the investigation.

- Comprehensible - The description used to represent the provenance of a piece of information shall be simple and be easy to understand in a human interpretable form. This is a particularly important characteristic when provenance is to be described in a court that may consist of non-technical people - the intended audience should always be taken into consideration.

A DEB records the provenance of the information stored within it, this could be acquired from a number of disparate sources, for example, a physical device or logical file. This raises a number of issues, especially if we are aiming to record information that complies with the attributes of 'good' provenance. For example if we store a file, what information should be recorded to define its origin?

When a user of a system saves a file to a hard disk then it is common to not only give the file a meaningful name, but also to store it in a folder (directory) where it can

easily be referenced and retrieved in the future. In addition to the file and folder name the user often knows the drive name/letter to which the file is saved. The drive is often referenced in the Microsoft environment by its drive letter. For example a user may know that a local hard disk is assigned the letter 'C' or the network drive may have an assigned drive letter of 'N'. Obviously this varies system by system and the user becomes accustomed to whatever drive letter is used on their system.

The assignment of that drive letter may be automatically allocated by the operating system. For example, if the system has a Microsoft Windows 98 operating system installed then the drive letter assignment is usually done by the order, type and number of partitions that are found within the partition table of the hard disk (*Microsoft 2004*). With Unix or more recent Microsoft operating systems the drive letter allocation may be slightly different (*Microsoft 2003*) and may even be configured by the user (*Microsoft 2007c*).

From the user's perspective, the drive letter followed by folder path and file name in the following form 'C:\My Documents\Hello.doc', is how the user routinely references their information. However, in the digital forensic world this often gets more complicated, but when presenting file provenance information to a court it is worth bearing in mind this is what the majority of computer users are familiar with.

The drive letter followed by folder path and file name, to the digital forensic specialist, are a logical description of the file location. Although it provides a unique (relative to the state of the system at that time), unambiguous and concise descriptor for the provenance from a non-technical users perspective, there are also other descriptors that may be used, for example the logical cluster numbers that the file occupies. This more technical definition records more information than the logical path and file name as it could also show if the file was fragmented in any way. This may be a significant piece of information that would be lost if only one form of provenance were recorded. Further, another unique, unambiguous and even more concise descriptor could be used, the physical sector locations that the file occupies could be recorded.

This leads to the questions; Which of these is best? Does it matter? Does one method mean the evidence is 'better' or has more integrity than another? Should we just record as many provenance descriptions as possible? It can be argued that in its own way each method meets the 'good' provenance criteria. It just depends upon the technical knowledge of the person trying to understand it. For example the general public, judge or legal professional is likely to be more familiar with a folder location than a more technical absolute disk sector or cluster reference. These other 'more technical' provenance descriptions may only complicate matters by introducing more technical vocabulary that actually detracts from and obscures the real information to be presented. In an ideal world we would record all provenance descriptions.

This would lead to multiple provenance definitions (keys) that can be used to sort, reference and catalogue information. The following example shows the keys for hard disk based information but the same principal could equally be applied to digital information obtained from other devices:

- Primary Provenential Key = Physical sector locations;
- Secondary Provenential Key = Logical cluster locations within a volume with the addition of an offset from the beginning of the physical device;
- Tertiary Provenential Key = Folder location specified from the root folder.

A DEB is capable of recording multiple provenance descriptions for a single entity stored in the bag file. These descriptions are stored in the index file and can be used to aid sorting and searching for information within the DEB or EU. The sequence in which the provenance descriptions are stored within the index file implies the key order (Primary -> Secondary -> Tertiary -> ...).

Mapping between different provenential keys is possible but only with additional information. For example, conversion from a physical sector number to logical cluster number requires partition offset and the number of sectors per cluster count.

3.4.2 Confidentiality

Confidentiality although important in an evidentiary context is usually not of primary concern as strict physical controls are usually in place to restrict who has access to the material. Thus access would usually only be available to personnel who have the necessary clearances and authorisation. Additionally, once evidence is placed in an evidence bag and sealed then casual access is denied without breaking the seal.

The DEB structure can be used to protect the information contained within the bag file from unauthorised or casual access. This is accomplished by obfuscating the contents by either compressing or encrypting the contents of the bag. This is analogous to protecting evidence in the physical world by the use of opaque evidence bags so that the contents cannot be seen until it is opened.

The protection that can be placed around digital evidence using cryptographic methods can be stronger than traditional evidence security. This would however require the implementation of key management mechanisms and policies for the issue and distribution of key material. As previously stated, evidence may have to kept for up to 30 years, this is a significant factor that has to be considered for not only the long term management of key material but also the effectiveness of encryption mechanism used.

3.4.3 Integrity

Integrity is the main security concern for digital evidence together with the ability to be able to prove that the information contained within the bag has not changed since it was seized (see requirements section 2.4 and 3.1.1). This is an area in which digital evidence differs from wet forensic sciences such as DNA and chemical analysis as these require that part of the original sample be used to conduct the analysis. These procedures must not contaminate the remaining original sample, however the original sample may have reduced volume. Alternatively, the properties of materials examined using these techniques may decay over time.

In the digital environment it could be quite trivial to modify a number of bytes of information in a manner that would be undetectable. Further, it is common for digital

storage media to deteriorate over a period of time. For example, the stability of information written to CD/DVD substrate material is known to have a limited shelf life (*OSTA 2003*). Even hard disk technology can be unreliable over a number of years but that could be due to physical attribute conditions such as hard disk bearing seizure, heads sticking to the platters or unsuitable environmental conditions (temperature and humidity). These are the main factors that have to be guarded against with regards to digital evidence integrity. The methods used are tried and trusted and involve the use of Cyclic Redundancy Checks (CRC) and cryptographic one-way hashes or a combination of the two. The DEB framework provides integrity assurance of the data contained within the bag file and additionally uses the same method to assure the integrity of the bag and index file components of the DEB. The integrity check values are generated by DEB aware applications at the time of DEB population. These methods do not preclude subsequent unauthorised modification of the data contained within the DEB by unscrupulous personnel, but does allow detection should accidental corruption occur. To prevent unauthorised modification a public key encryption scheme could be used, the details of how such a scheme would be incorporated into the DEB framework is an area for further work. The current DEB framework also assumes that the device is able to be read correctly without error.

Additionally the DEB framework when used with DEB aware and compliant applications can provide greater integrity assurance of evidence and results obtained due to the integral audit processes (see sections 3.4.5 and 3.4.6).

3.4.4 Availability

To analyse captured digital information the forensic analysis tool usually requires that the whole of the captured data be made available to the tool in order for analysis to commence. Therefore, this potentially imposes a number of limitations:

1) Sufficient fast access storage capacity is required to contain the whole image and also the data extracted from it: if the image is shared over a network, access time may be impaired;

2) Typically, only one system 'box' (shared memory processor) can be used to process the image at a time.

In order to mitigate these issues the DEB framework permits the distribution of EU component files in a structured manner; thus improving access performance and availability and also allowing a distributed or multi-processor environment to be utilised.

Similarly, multiple (disparate) forensic applications, each performing a different function, could operate on separate EUs. This allows the most appropriate analysis techniques to be applied to the information. For example, a keyword search to document files and graphical analysis techniques to picture/image files.

3.4.5 Continuity

Once evidence has been seized and its provenance recorded, it is exceedingly important to be able to track the continuity of who, where and when other persons took possession and held custody of the evidential material. This is accomplished in the traditional physical evidence handling environment by paper based property store records that record property movements. This is referenced using tags attached to the bags or containers that encapsulate the evidential material (see requirements of traditional evidence handling - section 3.1).

Should anyone wish to examine or perform analysis on the evidential material then the tamperproof seal has to be broken. This should be recorded in case notes and property store records, and once examination is complete then the evidence has to be resealed in another tamperproof container and its new seal number duly recorded. Typically if the physical seal has to be broken or the bag cut then the seal or bag would be placed within the new container thereby providing the audit trail of all custodians and containers that were used to hold the evidence.

DEBs have the ability to record continuity information in the tag file by the use of Tag Continuity Blocks (TCB). These are similar in nature to the continuity sections on a physical evidence tag, although the information in a DEB TCB may be more comprehensive.

3.4.6 Process

Current forensic applications allow the examiner to record and highlight (bookmark) relevant information but do not record the process that was undertaken during the investigation or provide any facilities to record the thought processes and justification for any actions taken by the examiner.

The DEB framework provides the ability to log the processes undertaken and the facility to record any notes or associated data the investigator deems necessary. This provides the necessary audit trail for other examiners to retrace the steps that were performed. This is quite a significant feature as it allows the development of optimal methods for case processing and provides the tools to identify if errors in process or procedure occurred. The process information is recorded in the DEB in a separate case note EU that is dedicated to the storage of process and contemporaneous notes. TCBs complement the process recording mechanism in case note EUs as they record application signatures and when accesses to the DEB contents were made.

A Digital Investigation Process Language (DIPL) was defined in (*Stephenson 2003*) and (*Stephenson 2004*) as a means to formally define and understand the processes undertaken as part of an investigation. DIPL is a LISP based language used to define the sequence of processes and actions undertaken as part of an investigation. A DIPL document of an investigation could be stored in a DEB.

3.4.7 Time

When conducting digital forensic examinations it is important to record the time when an event occurred for a number of reasons (see requirements defined in section 2.4):

1) to be able to ascertain the validity of evidence being examined;
2) as a point of reference when examining evidence;
3) timelining or event sequencing/ordering.

When evidence is seized it is common for the investigating officer to log the date and time on the tag attached to the material seized. It is common practice to use the time source that is currently to hand, as accuracy to within a minute or so is usually

adequate, however in a digital environment the resolution and accuracy of a timestamp may be crucial in correlating and corroborating information.

The recording of when particular events occurred in a digital environment is a function that most applications and systems perform automatically (*Farmer & Venema 2005*). The timestamp information is usually based upon the device's real-time clock or obtained from a network or radio time source. The resolution of these time sources may be a millisecond or less, but the accuracy of the time source is usually not important until there is a need to correlate information.

This seeming harmless discrepancy can cause major problems when correlating information from disparate sources or across different time zones, not withstanding the additional confusion that can be caused with daylight saving adjustments. There are a number of different time formats (*IETF 2002*), (*WETSTONE* 2007), that are encountered in digital environments which can lead to correlation difficulties due to the variability of resolution, accuracy of source and the fact that different epochs may be used.

The DEB framework stores date and time in UTC format. The source of the timestamp can be the real-time clock of the host system or some other trusted and verified source such as radio clock or network time service. It is good practice on behalf of the investigator to verify and record accurate time and time zone information when the investigation commences. The DEB framework supports this concept by permitting timestamp details to be recorded when the DEB is created and accessed by DEB aware applications. This enhances the provenance and continuity audit trails, thus providing confidence that the evidence was not changed or tampered with.

3.5 DEB Syntax Definition

This section defines the syntax used to implement a DEB based on the abstract components in section 3.3. The language used in this section shall be interpreted in accordance with RFC 2119 – 'Keywords in RFCs to Indicate Requirement Levels' (*IETF 1997*).

Figure 15 shows a detailed DEB structure diagram.

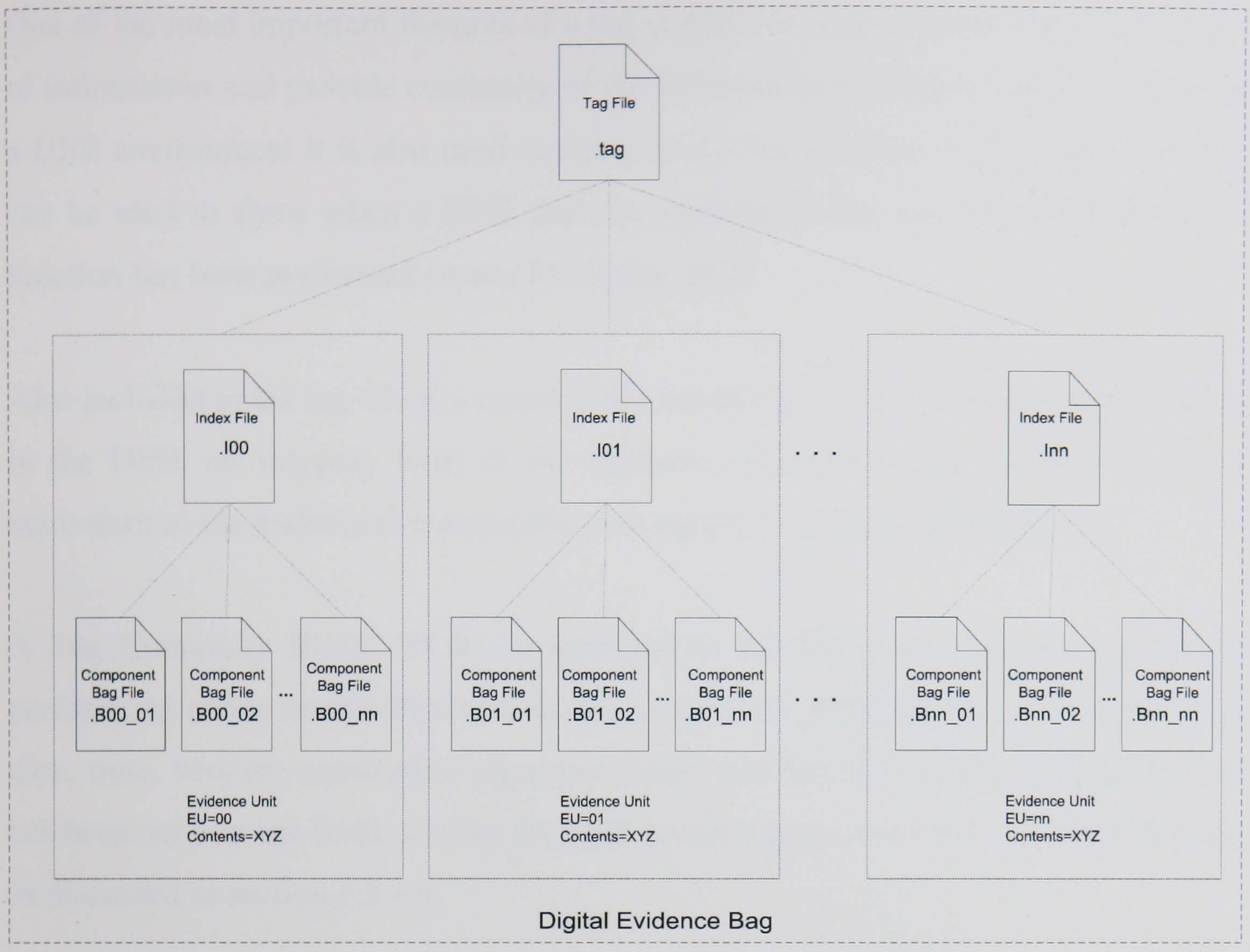

Figure 15 - Detailed DEB diagram

3.5.1 Tag File – General Information

The tag file is a plain text file containing the details of the evidence contained within the DEB. The information contained in this file is similar to that of a physical tag attached to evidence when it is seized.

Therefore it contains the following basic details:

- An evidence identifier or reference;
- The name of the person capturing the information;
- The date and time when the capture process started;
- A description of the contents of the DEB;
- Continuity information.

Some of these attributes are self explanatory in their meaning and content, but in the digital environment there is more scope to record information about the evidence and maintain its integrity.

One of the most important features of a tag is that it is used to record the provenance of information and provide continuity of the information contained within the bag. In a DEB environment it is also used in this context, but its scope is expanded in that it can be used to show when a DEB analysis application has used the DEB and what function has been performed on any EU in the DEB.

Also included in the tag file is a count of the number of EUs (.Inn and .Bnn file pairs) in the DEB, an integrity hash of the captured data and a Tag seal hash. This is equivalent to the traditional seal number and signature of the seizing officer.

A Tag Continuity Block (TCB) is appended to the file every time an application performs an action on the DEB as a whole, or part of a particular EU. This records the date, time, version, application signature (hash) and function of the application that has been used on the DEB. Ideally the DEB access process will have ACID properties as discussed in section 3.3.1.4.

In addition to this the DEB application will generate a new Tag seal hash that is calculated over the contents of the Tag file including the new TCB information.

One important component that gives the DEB the flexibility to hold information captured from a variety of sources (i.e. static, real-time, and selective) is the Meta-Tag structure used within the DEB header.

In certain circumstances it may be an advantage to compress or encrypt the contents of either a whole DEB or part (EU). This is done by specifying a content format type.

Evidence Unit fragments are also permitted to allow full device acquisition and each EU fragment to fit onto a backup media.

3.5.1.1 DEB Header

The DEB tag file shall start with the following entry:

```
[DEB Header vM.N]
```

Where M is the major version number and N is the minor version number. This allows an application parsing the DEB tag file to identify which DEB definition the file complies with.

Following the [DEB Header vM.N] line, additional information regarding the affiliation of the investigating officer and exhibit details shall be recorded. If any of the entries are not required then the line header shall be included with no following detail information:

```
[DEB Header v0.82]
Investigating Agency : <Organisation Name - mandatory>
Investigating Officer : <Investigator Name - mandatory>
Exhibit : <Exhibit Reference Number - mandatory>
Description : <Textual Description - mandatory>
Location : <Textual Description - mandatory>
Case Reference : <Case Reference Number - mandatory>
DEB Created Date & Time : <Timestamp - mandatory>
Host ID : <Host Identifier Description on which DEB is created e.g
Host name, IP Address - optional>
Password : <Device or host Password/PIN/Passphrase on which DEB is
created - optional>
```

3.5.1.2 Meta-Tags (MTs) – Index Format

In order to define the contents and format of the Index file the structure is defined in the Tag file. The structure definition is comprised of a series of Meta-Tags (MTs). The MTs are used to define both the sequence of fields and content type of the Index file.

In order to specify the format of an Index file the Tag file shall include the line:

```
Index Format : <Meta-Tags>
```

This should be the last line in the [DEB Header vM.N] section of the file if used for the whole DEB. An Index Format may also be specified for each EU.

The sequence that Meta-Tags appear in the Index Format entry is the order of the provenential keys. For example, the Primary provenential key is the first meta-tag, the Secondary provenential key is the second meta-tag and the Tertiary provenential key is the third meta-tag.

The index file holds information relating to the contents of the bag file.

3.5.1.3 Meta-Tag Definitions

<PS> = Physical Sector Number – contiguous runs of sectors, bytes per sector:nnn-nnn,nnn-nnn,…

<LCN> = Logical Cluster Number - contiguous runs of clusters, offset:sectors per cluster,nnn-nnn,nnn-nnn,…

<C> = command entered into Command Line Interface (CLI)

<F> = Filename and path

<Fx> = Filename extension/type

<Fa> = File Attributes (E.g. System, Read, Archive, Hidden)

<Fmft> = Master File Table (MFT) Index number

<Fls> = Logical File Size in bytes

<Fps> = Physical File Size occupied in bytes

<P> = Provenential Information

<Ds> = Data source (PDA) – RAM, ROM, Database (User Data), SIM, Handset

<Hmd5> = Hash MD5

<Hsha> = Hash SHA

<Rnum> = DEB Record Number

<Rlen> = Record Length

<Tmod> = Timestamp – last modified / completed

<Tacc> = Timestamp – last accessed

<Tcre> = Timestamp – created / started / executed

<Temo> = Timestamp Entry modified (NTFS)

<Tpacket> = Packet time

<Ddes> = Device descriptor

<Dman> = Device manufacturer

<Dmod> = Device model

<Dsn> = Device serial number

<Dcap> = Device capacity

<Dpin> = Device PIN, security access code, password

<Dsp> = Device service provider (phone)

<Raw> = Bag contents are raw/binary no structure

3.5.1.4 Extending Meta-Tags

The use of Meta-Tags can also be extended by permitting special 'short form' or 'alias' definitions for commonly used sets of Meta-Tags. This aids in keeping the Tag file clutter free and more easily understandable.

The following examples define special Meta-Tags for FAT and NTFS filing systems. It is common for a filing system to have a number of attributes associated with a file. The special Meta-Tags provide a mechanism for defining the expansions within a DEB Index Format definition.

<FAT> ≡ `F Fx Fa Tmod Tacc Tcre Fls Fps P Hmd5`

<NTFS> ≡ `F Fx Fa Tmod Tacc Tcre Temo Fls Fps P Hmd5`

3.5.1.5 Evidence Units (EUs)

The Evidence Unit definition shall immediately follow the DEB Header section and the Index Format specifiers.

This section of the Tag file shall commence with the following line:

```
[Evidence Units]
```

Information specific to each EU shall be defined in the [Evidence Units] section as follows:

EU=nn

<Format type definition>

where nn is the EU number

3.5.1.6 DEB or EU – information format types

This section defines the format type of information in either the whole DEB or EU part. The scope of each of these format types is dependant upon the placement of the format type specifier in the DEB. If the type specifier is in the DEB Header then the format applies to the whole DEB. If the type specifier is in the EU definition then the format applies to only that EU.

Evidence Unit 0 (EU0) is reserved for the storage of arbitrary case notes and the recording of process, see section 3.1.6. This can include text entries or associated digital metadata files, for example, documents or photographs etc.

3.5.1.7 Content Types

This defines the type of content that is contained in each EU. The definitions in this section specify an initial set of types. Further work is required to formalise and manage this set and permit the registration of new types. The practicalities of how this should be accomplished have not been considered in this work but there is precedence such as the definition of MIME types which is controlled by the Internet Assigned Numbers Authority (IANA) - http://www.iana.org/.

In order to specify the content type of the DEB as a whole or a particular Evidence Unit (EU) the tag file shall include the line:

```
ContentType-Sig=<File Signature1>,<File Signature2>…

Or

ContentType-Ext=<File Extension1>,<File Extension2>…

Or

ContentType-Cat=<Category Type>

Or

ContentType-Manual=<label>

Or

ContentType-CLI=<label>
<label> - Process List, Users, Network Config, Directory List etc.

Or

ContentType=Case Notes  - Contents are Case Notes, usually only used
by Evidence Unit 0

Or

ContentType-Format=<format>   -   Contents   are   another   image   file
format. For example, Raw, EnCase or AFF.

Or

ContentType-DEB   - Contents are another DEB. This is used if one DEB
encapsulates another.

Or
```

ContentType-DEV=<Device Type> - Valid types are Disk, Mobile Phone, PDA, Magnetic Swipe Card Reader/Writer (skimmer).

The <File Signature> shall be in the form \xFF\xFF\xFF... where FF is a series of hexadecimal byte values (magic number) at the start of a file.

The <File Extension> shall be the file extension or suffix used as part of the filename.

<Category Type> e.g. Documents, System Files (OS specific), Unallocated Space, Drive Config (Partition Table/ Filing System specific), Email, Encrypted, Internet History, Recent Documents and User Profile.

The <label> is descriptive text entered by the user.

3.5.1.8 Integrity Types

This defines the algorithm that is used throughout the DEB tag file for integrity assurance purposes. Each DEB shall use one or more algorithms.

In order to specify the integrity algorithm used throughout the Tag file the DEB header shall include the line:

Integrity Type : <Type1> <Type2>...

The currently defined types are:

MD5 – Message Digest 5 algorithm – RFC 1321

SHA1 – Secure Hash Algorithm – RFC 3174

Once the integrity algorithm type is defined the following specifies the Index and Bag file integrity hashes:

IndexHash=<Hash> - Index file hash, created when index file is closed.

BagHash=<Hash> - Bag file hash, created when bag file is closed.

Tag File Hash : <Hash> - Tag file hash, created when tag file is closed

3.5.1.9 Compression Types

This defines the algorithm that is used throughout the DEB or EU for data compression.

In order to specify the compression algorithm used the tag file shall include the line:
`Compression=<Type>`

If the compression type definition is in the DEB Header then the compression type used applies to all the data in every bag file defined in the DEB. If the compression type definition is in the EU block then the compression type used applies to all the data in its corresponding EU bag file only.

The currently defined types are:
NONE – No compression used (default)
ZLIB_0 – bag file contents are compressed using the ZLIB fastest algorithm – RFC 1950
ZLIB_1 – bag file contents are compressed using the ZLIB fast algorithm – RFC 1950
ZLIB_2 – bag file contents are compressed using the ZLIB default algorithm – RFC 1950
ZLIB_3 – bag file contents are compressed using the ZLIB maximum compression, slowest algorithm – RFC 1950

3.5.1.10 Encryption Types

This defines the encryption algorithm that is used throughout the DEB or EU.

In order to specify the encryption algorithm used the tag file shall include the line:
`Encryption=<Type>`

If the encryption type definition is in the DEB Header then the encryption type used applies to all the data in every bag file defined in the DEB. If the encryption type definition is in the EU block then the encryption type used applies to all the data in its corresponding EU bag file only.

The currently defined types are:

NONE – No encryption used (default).

DES – FIPS-46-1 US National Bureau of Standards, "Data Encryption Standard". Federal Information Processing Standard (FIPS) Publication 46-1, January 1988.

AES - National Institute of Standards and Technology, "Specification for the Advanced Encryption Standard (AES)" FIPS 197. November 26, 2001.

3.5.1.11 Timestamps

All timestamps used in a DEB shall be of the format defined in ISO 8601, RFC3339, e.g.:

```
YYYY-MM-DDThh:mm:ss.sTZD (eg 1997-07-16T19:20:30.45+01:00)
```

where:

```
     YYYY = four-digit year
     MM   = two-digit month (01=January, etc.)
     DD   = two-digit day of month (01 through 31)
     T    = date/time separator
     hh   = two digits of hour (00 through 23) (am/pm NOT allowed)
     mm   = two digits of minute (00 through 59)
     ss   = two digits of second (00 through 59)
     s    = one or more digits representing a decimal fraction of a
second
     TZD  = time zone designator (Z or +hh:mm or -hh:mm)
```

3.5.1.12 DEB Footer

When the DEB Tag file is closed after its initial creation it shall end with the following entry:

```
[DEB Footer]
Evidence Units in DEB : <number of Evidence Units in DEB>
DEB Closed Date & Time : <Timestamp>
Tag File Hash : <Hash>
```

3.5.1.13 Tag Continuity Blocks (TCBs)

The purpose of the Tag Continuity Block (TCB) is to provide an audit trail of applications that have accessed the DEB or an EU within the DEB.

All applications requiring access to a DEB or EU shall do so through the Tag file. In doing so the Tag file shall be updated by appending a TCB to the end of the file. The accessing application identifier and version number are supplied by the calling application.

Each TCB shall be of the format:

```
[TCB]
Date & Time : <Timestamp>
Application ID : <App ID>
Application Version : <Application version identifier>
Application Signature : <Hash of Application>
Application Function : <App description/function>
Host ID : <Host Identifier Description e.g Host name, IP Address>
DEB Components Accessed : <ALL | EU number list>
Tag File Hash : <Hash>
```

3.5.2 Index File

This is a plain text file which is tab delimited.

.Inn (Index) file – a list detailing the contents of the corresponding .Bnn file. The index could contain details such as a list of filenames, folder paths, and timestamp information relating to the contents of the digital information in the bag. Alternatively it could contain details of the physical device, for example the make, model and serial number of the device from which digital evidence has been obtained.

Its exact format and structure is defined in the Meta-tag information in the .tag file.

3.5.3 Bag File

This file contains the evidential data/information.

The .Bnn (Bag) file contains the actual evidence captured and may be either:

- Raw binary information. For example, from a raw device capture application such as 'dd' or an EnCase evidence file 'E01' format data;

- Arbitrary collection of files. For example a manual selection of files from logical volume acquisition;

- Categorized / grouped collection of files. For example, one bag containing all text files, another containing all MS word documents, or another containing all JPEG graphics files etc;

- Structured binary information. For example, from network packet capture.

- An alternative approach is to create one Bag file per actual file acquired. For example, one JPEG file into one Bag file. This results in very large Tag files and is probably not the most practical or efficient way to store information.

3.6 DEB API Implementation

In order to create, open and update a Digital Evidence Bag, a number of basic functions are required to manipulate and access the component parts. The following Pascal definitions define the DEB API. Pascal was chosen due to its lingua franca properties. The functions belong in two main classes, those for DEB creation and DEB access. Additionally a number of record types are defined to group sets of information that are required to be passed into functions or used to maintain state of a DEB object. Three have been defined, they are:

- Case Details record;
- Evidence Unit Details record;
- Bag Object Summary Information record.

The definition of the three record types now follows:

```
// Case details record
TCaseDetails = record
    sInvAgency : string;                // Investigating Agency
    sInvOfficer : string;               // Investigating Officer
    sExhibitNumber : string;            // Case unique Exhibit Number
    sDescription : string;              // Brief Description of contents of DEB
    sLocation : string;                 // Location where evidence was seized
    sCaseReferenceNumber : string;      // Case Reference Number
    sHostID : string;                   // Optional: Host Identifier that evidence was
                                        // acquired on
    sPassword : string;                 // Optional: Password/PIN/Pass phrase required to
                                        // access
                                        // device/host
    sNotes : string;                    // Optional: Additional text case notes
    sIndexFormat : string;              // Optional: DEB Index Format, may be specified per
                                        // EU
end;
```

```
// EU details record
 TEUDetails = record
    iEUNum : integer;           // EU Number
    sIndexHash : string;        // Index File hash
    sBagHash : string;          // Bag File hash
    sContentType : string;      // EU Content Type description
    sEUIndexFormat : string;    // EU Index Format if specified per EU
    iFpsColumn : integer;       // store column in Index Format that contains physical size
    iFlsColumn : integer;       // store column in Index Format that contains logical size
    iFColumn : integer;         // store column in Index Format that contains FileName
    iPColumn : integer;         // store column in Index Format that contains Provenance
    iCColumn : integer;         // store column in Index Format that contains Command Line
                                // Input
    iHashColumn : integer;      // store column in Index Format that contains Hash
end;
```

```
// Bag Object Summary Information
  TBagObj = record
     sF : string;                    // F = FileName
     sP : string;                    // P = Provenance
     sC : string;                    // C = Command Line Input
     sHash : string;                 // Hash
     lFls : Longint;                 // Fls = Logical File Size
     lFps : Longint;                 // Fps = Physical File Size
     lBagFileOffset : Longint;       // Offset into bag file to start of selected object
  end;
```

3.6.1 DEB Creation functions

The functions that are required to create a DEB must be able to create and populate the tag file and then be used to add additional Evidence Units as required by the particular application. Functionality must also be provided to fill the bag files with different types of data (e.g. binary or textual) from differing sources (e.g. file or memory buffer). One DEB may be created at a time, it is not envisaged that a creation application would be required to simultaneously create and populate multiple DEBs.

```
DEBCreate(      sAppVersion : string;
                sCopyright : string;
                CaseDetailsIn : TCaseDetails;
                slCaseFiles : TStringList;
                sImageDestinationPath : string;
                tCompressionAlgorithm : TcompressionAlgorithm;
                tEncryptionAlgorithm : TencryptionAlgorithm;
                tIntegrityAlgorithm : TintegrityAlgorithm ) : boolean;
```

```
Purpose
     This is the DEB class constructor that creates a DEB tag file
and populates the DEB header with case detail information, DEB
creation application detail. The return value indicates success
(true) or failure (false).
```

Function

CreateNewEvidenceUnit(sContentType : string) : integer;

Purpose

To create a new Evidence Unit(Index and Bag files) within the currently open DEB. This appends new EU information to the tag file. The return value is the number of the EU created (greater than or equal to zero), a negative value indicates an error.

Function

WriteEvidenceDataBuffer(iEUNum : integer;
byteBuf : Array of Byte;
iBufSize : integer) : integer;

Purpose

To write information to the specified Evidence Unit. The return value indicates the number of bytes actually written, a negative value indicates an error.

Function

function AddCaseIndexEntry(iEUNum : integer;
sEntryType : string;
iEntrySize : int64;
sEntryHash : string) : integer;

Purpose

To add a new entry to the specified EU Index file. The return value indicates the index number of the entry added, a negative value indicates an error.

Function

CloseEvidenceUnit(iEUNum : integer;) : boolean;

Purpose

This closes the specified Evidence Unit. This involves generating an integrity hash of the contents of the Index and Bag files. These values are then written to the Tag file. The return value indicates success (true) or failure (false).

Function

DEBClose(): boolean;

Purpose

This closes the DEB Tag and any open Evidence Unit files. If an EU is open an integrity hash of the contents of the Index and Bag files is generated and written to the Tag file. An integrity hash of the Tag file is then generated and a DEB Footer is then appended to the Tag file. The return value indicates success (true) or failure (false).

3.6.2 DEB Access functions

The functions that are required for DEB access have to be able to open an existing DEB and append continuity information to the Tag file. The Tag file has to be parsed in order to determine the quantity, content type and format of each Evidence Unit. Functionality also has to be provided in order to locate and access information within the Bag file of a specified EU. The information appended to the Tag file provides the basic mechanism to record the processes that have been performed on the DEB.

Additionally the detection of corruption or integrity failures in the structure of the DEB should be detected. This could be due to a number of factors such as:

- media errors;
- deliberate and malicious changes to the contents of the DEB;
- DEB creation application terminating abnormally, resulting in a partially populated and/or incorrectly closed DEB.

If an error is detected when a DEB is opened and accessed this could initiate some kind of error correction or automated repair process. The quantity and location of errors detected in the DEB structure will affect the possibility and effectiveness of any such repair. This would also necessitate recording the process and functions performed in order to correct the errors. How this would be achieved is an area for further consideration.

OpenDEB(sTagFileName : string; sAppID : string; sAppVer : string; sFunctionDesc : string): boolean;

Purpose

This is the DEB class constructor that Opens a DEB, parses the tag file to determine EU information and appends a TCB to the tag file. The application name, version and a short textual description should also be supplied. These details are included in the TCB. The calling Application Signature (hash) is calculated and written to the TCB along with the current system timestamp. The return value indicates success (true) or failure (false).

Function

GetNumEUs() : integer;

Purpose

The return value indicates how many Evidence Units are in the current DEB, a negative value indicates an error.

Function

GetEUContentType(iEUNum : integer) : string;

Purpose

The return string indicates the content type description of a particular Evidence Unit, a NULL return indicates an error.

Function

CloseEU(iEUNum : integer) : boolean;

Purpose

To close a particular Evidence Unit i.e. Index and Bag files. The return value indicates success (true) or failure (false).

Function

GetEUIndexFormat(iEUNum : integer) : string;

Purpose

The return string shows the Index Format definition of a particular Evidence Unit, a NULL return indicates an error.

Function

OpenEU(iEUNum : integer) : boolean;

Purpose

To open a particular Evidence Unit i.e. Index and Bag files. The system state of the currently open EU is maintained internally within DEB function. The return value indicates success (true) or failure (false).

Function

GetIndexEntryFirst(iEUNum : integer) : string;

Purpose

The return string shows the first index entry from the specified Evidence Unit Index file, a NULL return indicates an error.

Function

GetIndexEntryNext(iEUNum : integer) : string;

Purpose

The return string shows the next index entry from the specified Evidence Unit Index file, a NULL return indicates an error.

Function

GetBagEntrySummaryDetails(iEUNum : integer; iIndexNum : integer) : TBagObj;

Purpose

To get the Bag Object Summary Information record of a particular Index file entry number.

Function

```
GetBagData(   iEUNum : integer;
              iIndexNum : integer;
              lOffset : Longint;
              Buf : Pointer;
              iBufSize : integer) : integer;
```

Purpose

To return a pointer to the data contained in a bag file referenced by a particular Index file entry or offset within that entry. The return value indicates number of bytes read from the bag, a negative value indicates an error.

3.6.3 Alternatives to using the DEB API

Ideally, information stored in a DEB should be accessed via applications that implement the DEB API and are therefore capable of maintaining the continuity and audit trail. Enforcing DEB access policy has not been considered in this work and is an area to be considered in future DEB development. Using the API functions to access the contents of a DEB maintains the integrity of the DEB as the correct protocols are used and the DEB is 'resealed' correctly once evidence population, examination or analysis has been performed (see section 3.3.1.4 discussing the ACID properties). This could lead to the development of 'certified' DEB applications which can be verified that the DEB API has been implemented correctly and even the registration of certified applications. Additionally DEB hardware devices could be developed; this may assist with the confidentiality and key management issues identified earlier. This would be of particular use when there is a requirement to distribute evidential material to the defence, international investigations (cross jurisdiction) or other agencies e.g. the National Technical Assistance Centre (NTAC) for special processing.

If the DEB API is not used then it may be possible to access information stored in the Bag file directly; however there are a number of issues that should be borne in mind when using a direct access method:

- Difficulty in determining which bag files to open – This would require manual parsing of the tag file to determine the content type of each evidence unit and therefore which bag file set to open.

- Manual checking required to verify the integrity of the bag file set – Once the bag file of interest has been identified a manual comparison of the integrity hashes of the bag file contents with the signature stored in the tag file should be performed.

Until the DEB format and DEB aware analysis applications are widely adopted accessing the contents of a bag file directly may permit current forensic toolsets to be used. This would not be recommended, however it does permit a certain degree of backwards compatibility and does not negate the financial outlay required for purchase of the toolsets.

4 DEB Toolkit Implementation and Framework Demonstrators

Ideally, DEBs should only be accessed by 'DEB aware' applications i.e. those that understand the structure of the DEB and therefore can update and maintain the continuity of the information stored within the DEB (see section 3.6.3 for discussion on alternatives for non DEB aware applications). DEB aware applications fall into one of two categories; creation tools or access tools. The operational scenarios that these are used in place differing demands on the user interface and complexity of the particular application, not to mention the vast range of disparate devices that information may be obtained from.

To accommodate these differing scenarios a toolkit has been developed that allows for the creation and subsequent access of a DEB. The toolkit has been written in the Delphi Pascal programming language. It was felt important to use an environment that permitted development on a Microsoft Windows platform because the majority of digital forensic investigators are more familiar with that operating system and using applications on that system. Delphi is an ideal tool for this purpose as the code is relatively easy to understand and implement a graphical user interface to an application.

The translation of the core functionality into other programming languages and operating systems should be relatively trivial as the DEB tag and index files are text based. The majority of common programming languages provide basic string input, output and manipulation routines that can support the requirements of the DEB framework.

The following sections discuss a set of applications for DEB creation, viewer and logging applications that were created to implement the DEB framework. All applications use a common library implemented as a Delphi unit. The 'DEB.pas' unit contains public functions to create, open and manipulate the various components of a DEB.

4.1 DEB Creation Tool Demonstrator

Typically, when a digital device is seized a form of imaging or copying of the storage media within the device is performed. DEB creation and imaging tools initially have to create the tag file and record the provenance information of who, where and when the capture commenced. Ideally these tools should be easy to use because if mistakes are made it could result in the potential for evidence being destroyed, changed, overwritten, or the opportunity missed to seize the information within a live environment.

The DEB tag file is built sequentially; Header, EU descriptions and DEB Footer. The number and content type of the EUs is determined by the configuration and functionality afforded by the particular acquisition application. This in turn determines the 'Index Format' definition used within the entire DEB or particular EUs.

At the completion of the acquisition process the EU (index and bag file) integrity hash values are calculated and finally the tag file integrity hash is calculated and written to the tag file. This effectively seals the DEB until analysis is commenced.

Figure 16 shows a screen capture of the DEB Writer – API Demonstrator. This application may be used to create a basic DEB consisting of an arbitrary number of Evidence Units. The data written to the bag file using this application can be either text strings or a buffer of binary data. When used in a 'real' application the bag would be populated with data obtained from a device capture, files, command line input or any other digital source.

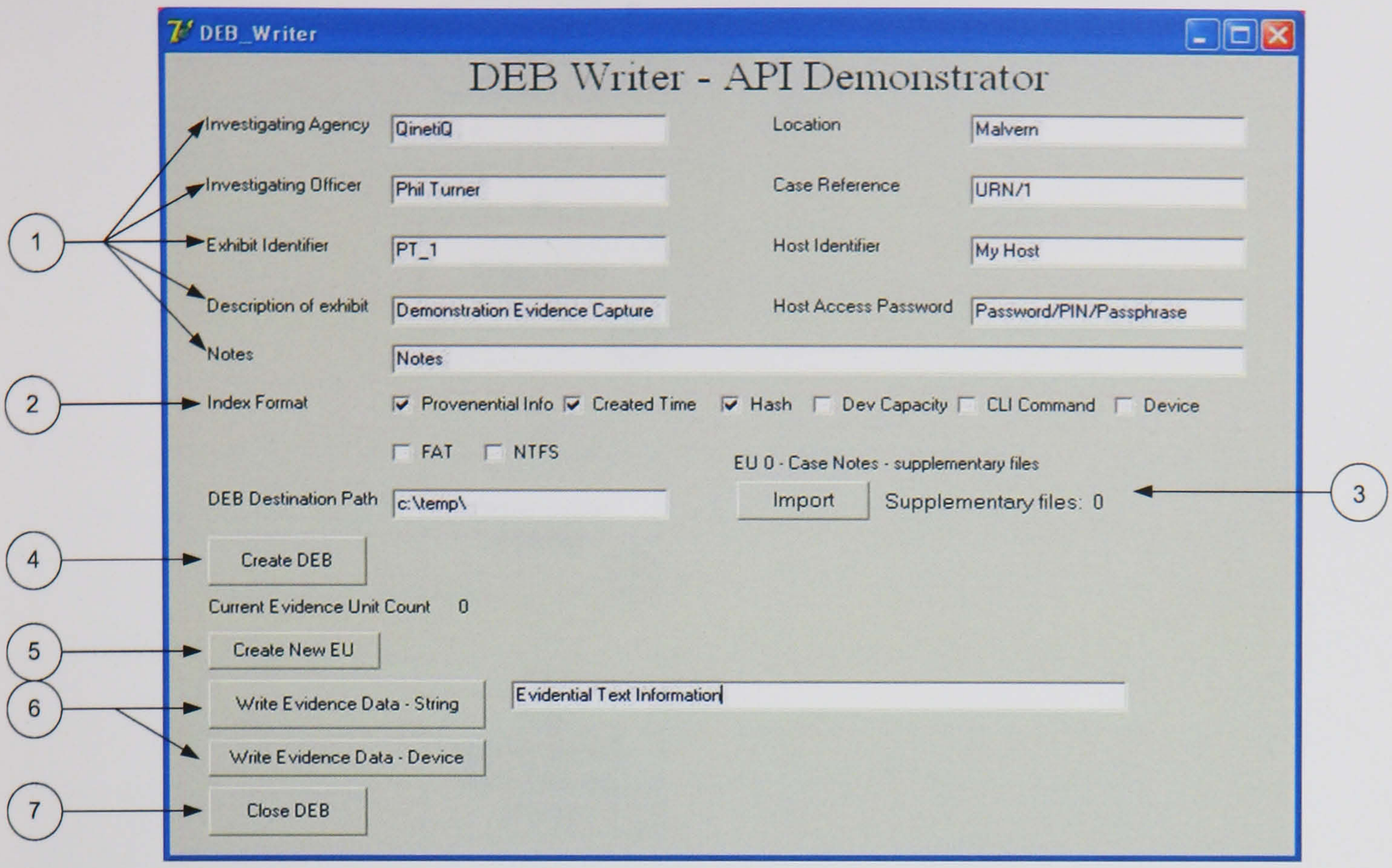

Figure 16 - DEB Writer API demonstrator

The numbered sections (1 to 7) should be read in conjunction with the application flow diagram and correspond to the sequence of operations shown in Figure 17.

Each step in the flow diagram perform the following functions:

1) The case details are entered by the user, this is the information held in a 'Case Details' record and is comprised of the following:

- Investigating Agency;
- Investigating Officer;
- Case unique Exhibit Identifier;
- a Brief Description of contents of DEB/ exhibit captured;
- the Location where evidence was seized;
- a Case Reference Identifier;

and the following optional details:

- Host Identifier that evidence was acquired on;
- Password/PIN/Pass phrase required to access device/host;
- Additional text case notes.

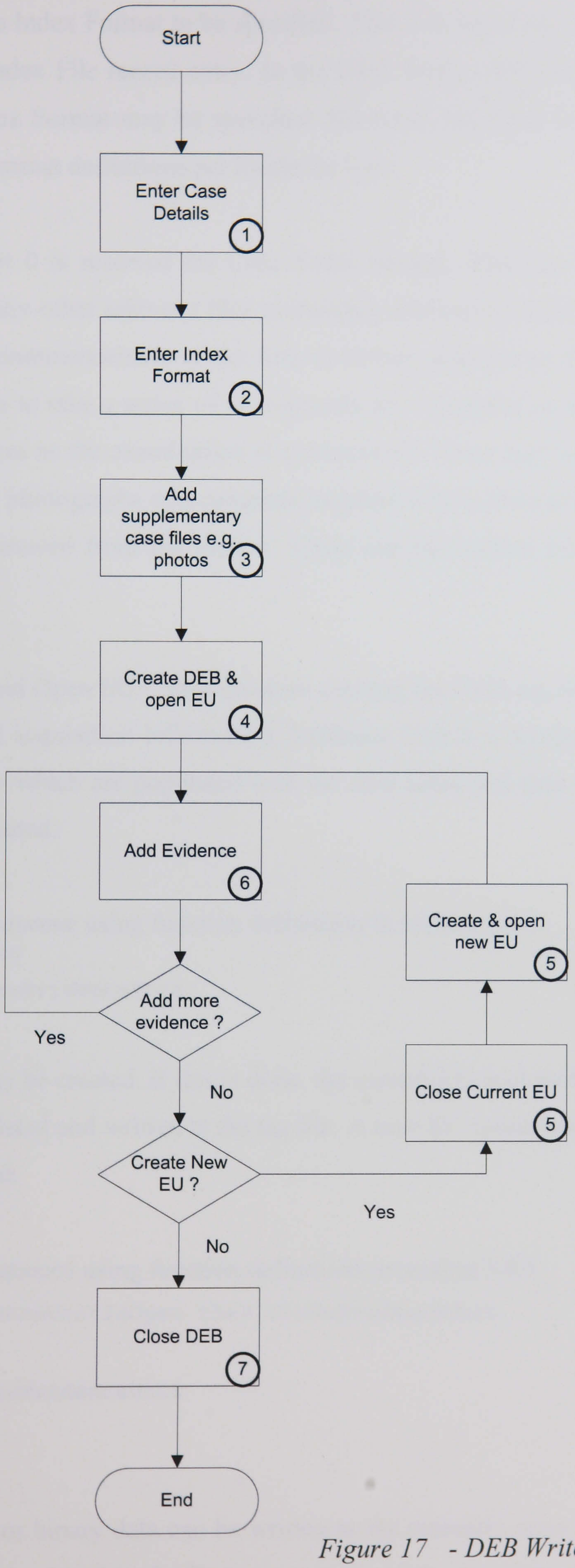

Figure 17 - DEB Writer Flow diagram

2) Allows for the Index Format to be specified. This is a list of the information that is stored in each Index File record entry. In the DEB Writer API Demonstrator only a global DEB Index Format may be specified. However, the DEB format does permit separate Index Format definitions per Evidence Unit.

3) Evidence Unit 0 is reserved for Case Notes storage. This can be documents or photographs or any other arbitrary files containing information relating to the case or exhibit prior to commencement of the data or device acquisition. For example, it is common practice to take a series of photographs of a computer or device illustrating its notable features as the examination is commenced. These may include front, rear, side and internal photographs of a computer together with a photograph of the storage media device removed from the system. These can be marked for inclusion in the DEB.

4) Create DEB and Open EU – This involves creating the DEB tag file and writing the DEB header and acquisition information. Evidence Unit 0 is comprised of an Index and Bag file pair which are populated with the case notes and then closed. Evidence Unit 1 is then created.

Function Call Sequence using function definitions in section 3.6.1:

```
DEBCreate

CreateNewEvidenceUnit
```

5) A new EU may be created. If this is done, the current EU is closed and a hash of its contents is calculated and written to the tag file. A new EU comprised of an Index and Bag file is created.

Function Call Sequence using function definitions in section 3.6.1:

```
If EvidenceUnitIsOpen then CloseEvidenceUnit
else
CreateNewEvidenceUnit
```

6) Either textual or binary data can be written to the currently open EU. Each time a new item of data is placed in the Bag a new index entry is written. The exact format

of the entry depends upon the Index Format specified; however, this would typically include provenential information, a timestamp of when the data was captured, the size of information acquired and an integrity hash of the data.

Function Call Sequence for binary or textual information using function definitions in section 3.6.1:

```
WriteEvidenceDataBuffer
AddCaseIndexEntry
```

7) DEB closed. This involves closing the current EU and calculating a hash of its Index and Bag file contents. These details are written to the Tag file and then a hash of the tag file is calculated and a DEB Footer written to the Tag file before closure. This acts as the DEB seal that is synonymous with the physical evidence seal attached to an evidence bag.

Function Call Sequence using function definitions in section 3.6.1:

```
If EvidenceUnitIsOpen then CloseEvidenceUnit
DEBClose
```

4.2 DEB Access/Viewer Tool Demonstrator

Once digital information has been captured and a DEB created, further investigation and examination tasks are usually required to discover further facts and correlate information. These analysis tasks may themselves output further distilled information, the provenance of which should also be recorded. These analysis tools can either extract information from the original DEB to a new DEB or build a new EU appended to the existing DEB.

In either scenario, to be DEB aware, an application must access a DEB by first parsing the tag file. This allows the application to determine the number and content type of EUs and verify their integrity hashes.

The analysis tool then appends a TCB to the end of the tag file detailing the EU accessed, the provided description of the function that the analysis tool performs, and

when the DEB was accessed. On completion of the analysis a new tag seal hash value should be generated to reseal the DEB.

The demonstrator currently writes the TCB when the DEB is opened (a single operation). The TCB update process requires further definition in order that a TCB can successfully record when a DEB is both opened and closed (two phase operation). This permits the subsequent detection of an access application process failure and can also be used to time the duration of a particular process.

Figure 18 shows a screen capture of the DEB Viewer application. This application may be used to open a DEB and catalogue the Evidence Units and their content type (4). An Evidence Unit may be selected and then the corresponding Index file (5) is parsed. An Index entry may then be selected and the data that it references is then viewed.

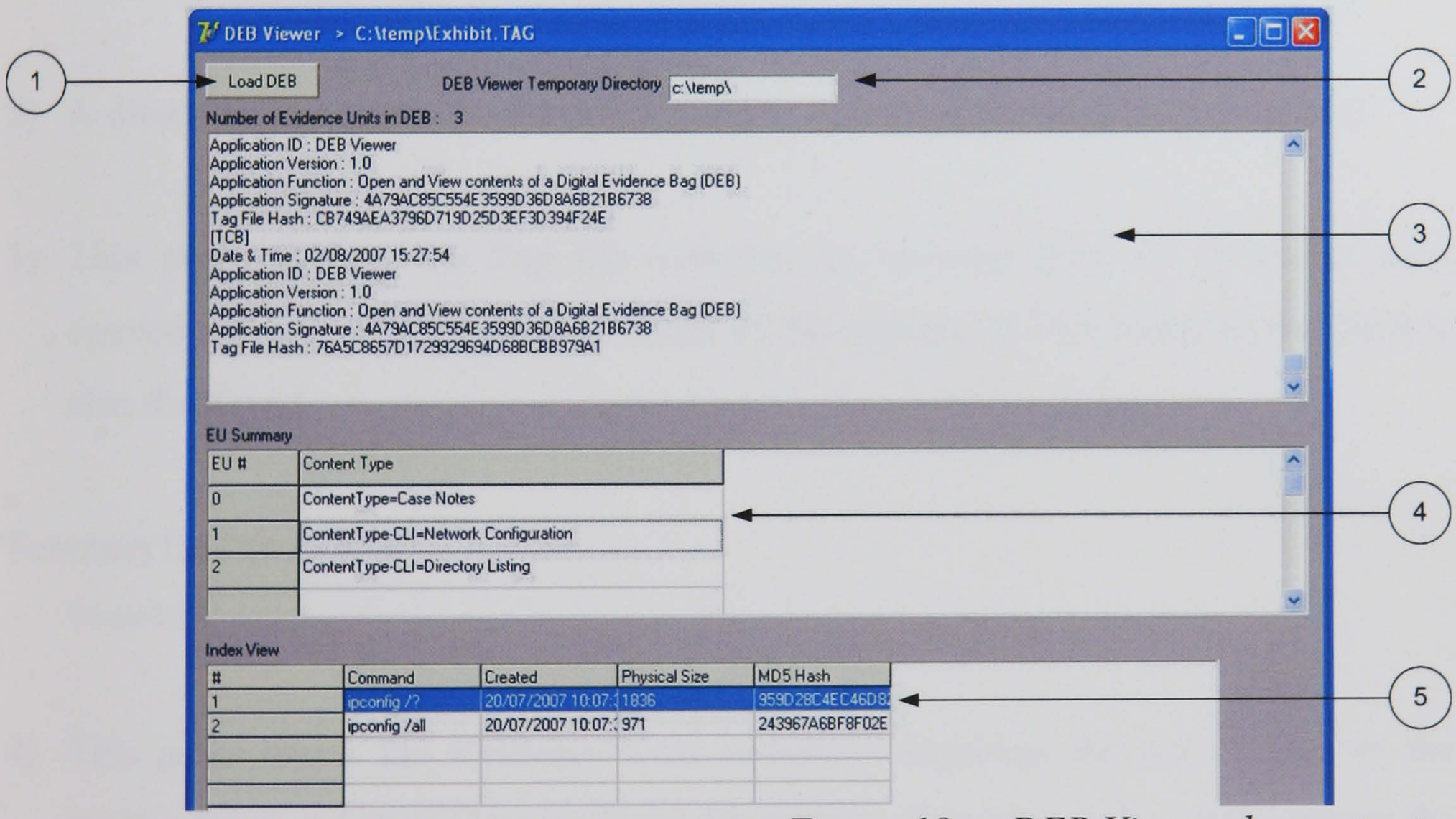

Figure 18 - DEB Viewer demonstrator

Each of the items (1 through 5) marked on Figure 18 is now discussed and an explanation of the function and API procedures called is given.

1) Load DEB – When a DEB is selected for loading into the DEB Viewer the application title bar is updated to show the name of the DEB selected. The tag file is parsed sequentially for the following:

Index Format - this is stored as the global definition that may be overridden by that specified in an individual Evidence Unit;

Evidence Units - each EU record is parsed to extract the content type. Index and Bag file integrity hashes. These are stored in an EU Details record;

DEB Footer is parsed to verify that the EU count matches the number of EU records parsed.

Once the tag file has been successfully parsed a Tag Continuity Block (TCB) is appended to the end of the Tag file that records the timestamp of when the DEB was opened, the Application Identifier, signature and description of the utility that is being used.

Function Call Sequence using function definitions in section 3.6.2:

```
OpenDEB
```

2) A directory can be set for the application to export information from the DEB.

3) This pane displays the Tag file contents for viewing after the DEB has been opened and a TCB appended. A count of the number of EUs found in the DEB is also displayed.

Function Call Sequence:

```
ViewTag
```

4) This pane shows the Evidence Unit summary depicting the list of EUs in the current DEB and their Content Type description. Selecting an EU entry causes the EU Index to be parsed.

Function Call Sequence using function definitions in section 3.6.2:

```
GetNumEUs
GetEUContentType
```

5) This pane is used to show the contents of the selected EU Index entry. As each Index entry is read an internal table is built to permit rapid access to the evidential data stored in the bag file. Selecting an Index entry causes another window to be

displayed that shows the data (in the form of a hexadecimal view) the entry relates to. The example shown below is for the command line application 'ipconfig /all', an example of the hexadecimal view window is shown in Figure 19.

Function Call Sequence - On selecting an EU:

```
GetEUIndexFormat
OpenEU
GetIndexEntryFirst
GetIndexEntryNext
```

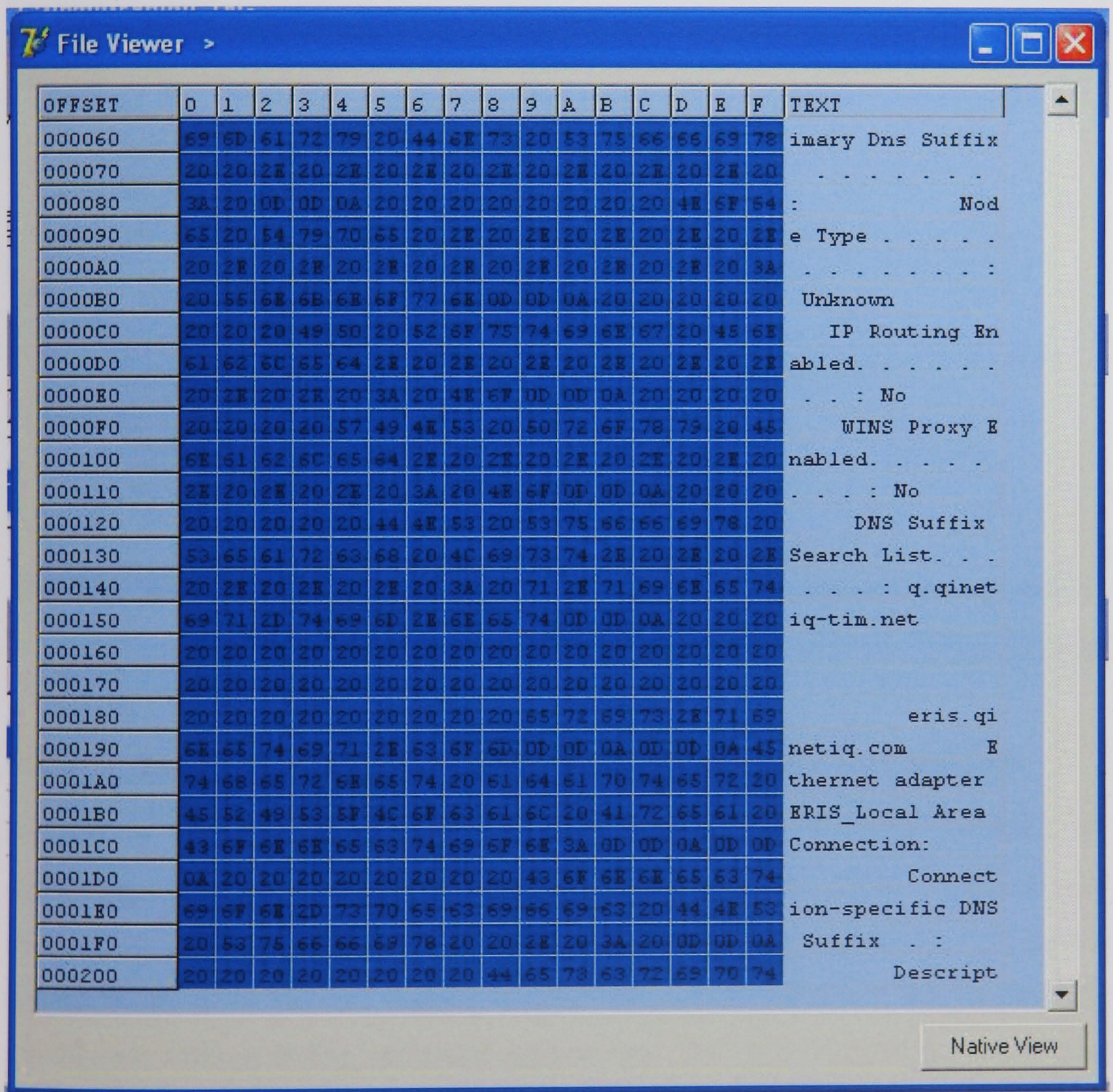

Figure 19 - Hexadecimal display of DEB Viewer demonstrator

Function Call – To extract data from the Bag file

```
GetBagData
```

6) In addition to displaying the selected entry in a hexadecimal view it is possible to view the entry in its native view. The default viewer in this application for text-based data is Microsoft's Notepad application. An example of the native view output shown in Figure 20.

Function Call Sequence – Native View

```
GetBagEntrySummaryDetails
GetBagData
```

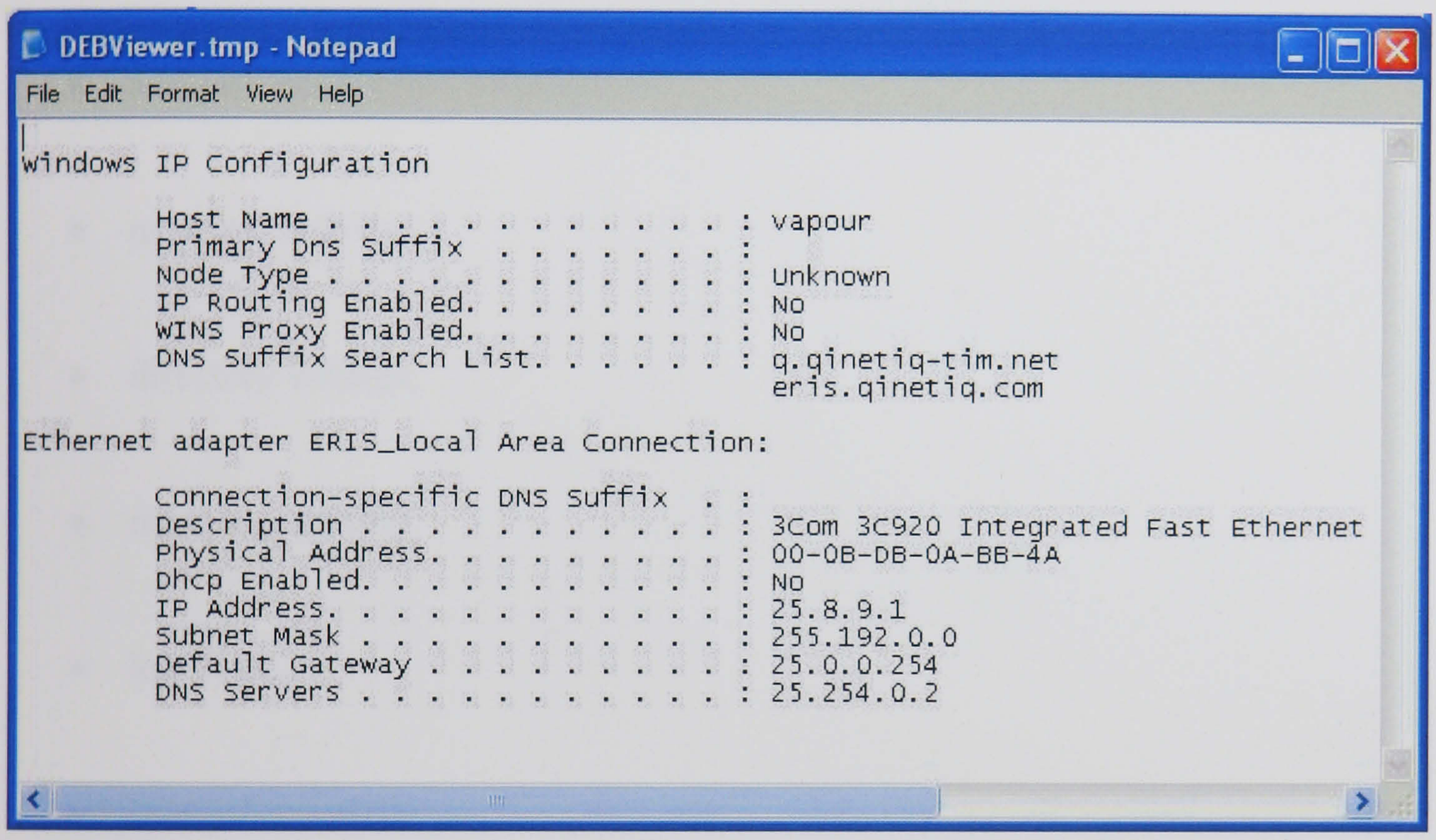

Figure 20 - Native View output example

4.3 Enhancing Current Applications - DEB Command Line Wrapper

As part of a digital investigation it is common to use the logs generated by applications that are already installed on a system or to utilise command line applications to capture a snapshot of current system information. Ideally an operating system would have forensic logging capabilities built-in. However, this is currently not the case without adding additional applications. The DEB framework and API can be used to provide an enhanced logging and command line capture facility for currently installed applications.

The DEB command line wrapper application provides an interface to execute any command line application and capture the output in a DEB. The application initially

creates a DEB and populates the DEB with case or incident reference information. The Command Line Application Wrapper is a variant of the DEB creation tool and utilises the same function call sequence described in section 4.1. Command line application tools and utilities can then be executed in an arbitrary order to record:

- system state;
- process lists;
- network connection information;
- network activity;
- directory listings;
- memory capture;
- log files;
- user information.

Figure 21 illustrates how these various types of information can be segregated into various EUs within a DEB. The division of information into separate EUs is an arbitrary decision made by the investigator. However, following a methodical process can permit each EU to contain similar types of information. The command and any associated parameters are stored in the Index file and the generated output resulting from command execution is recorded in the Bag file.

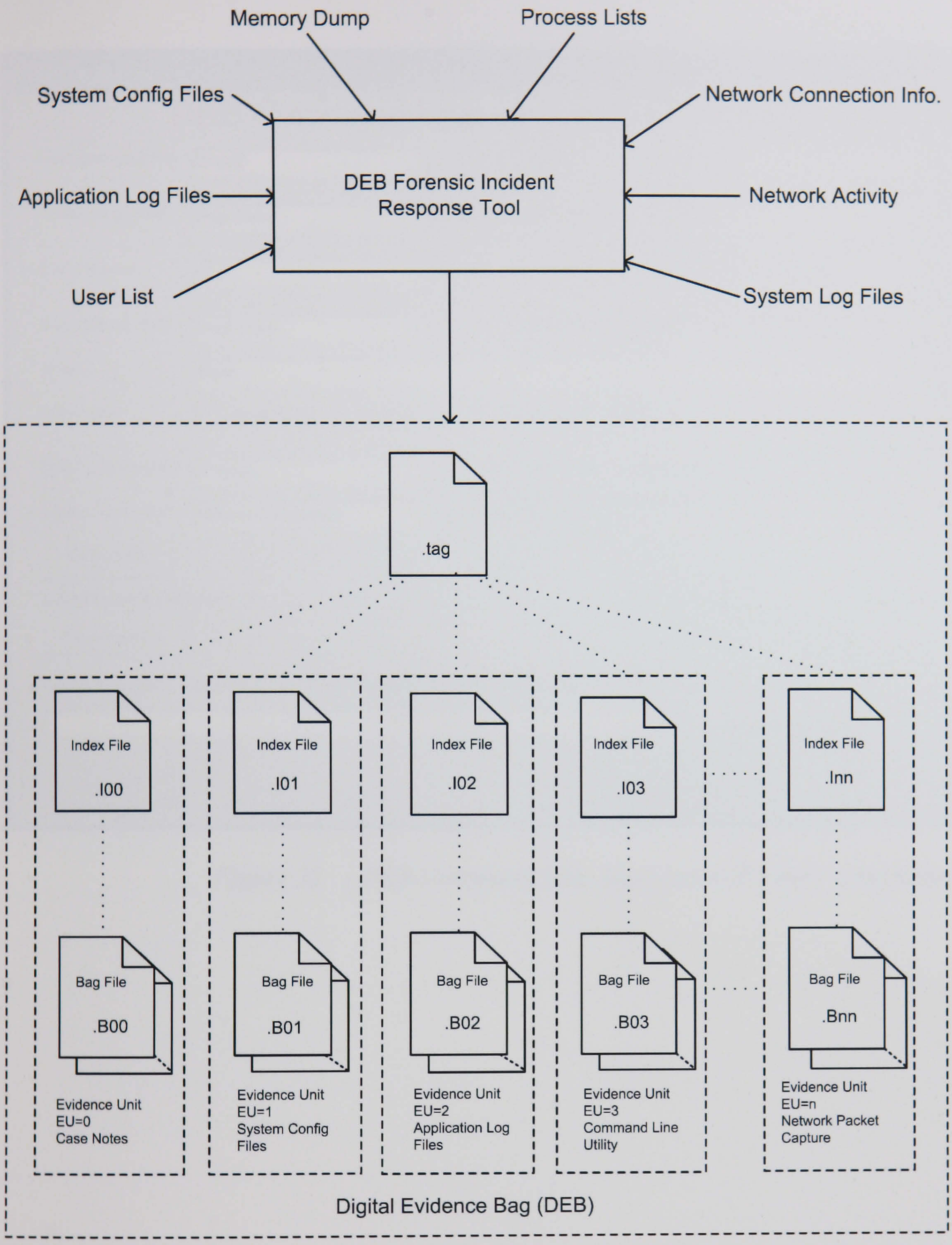

Figure 21 - DEB Information segregation

Figure 22 is a screen capture of the DEB Command-Line Application Wrapper utility that can be used to acquire information from the command line into a DEB.

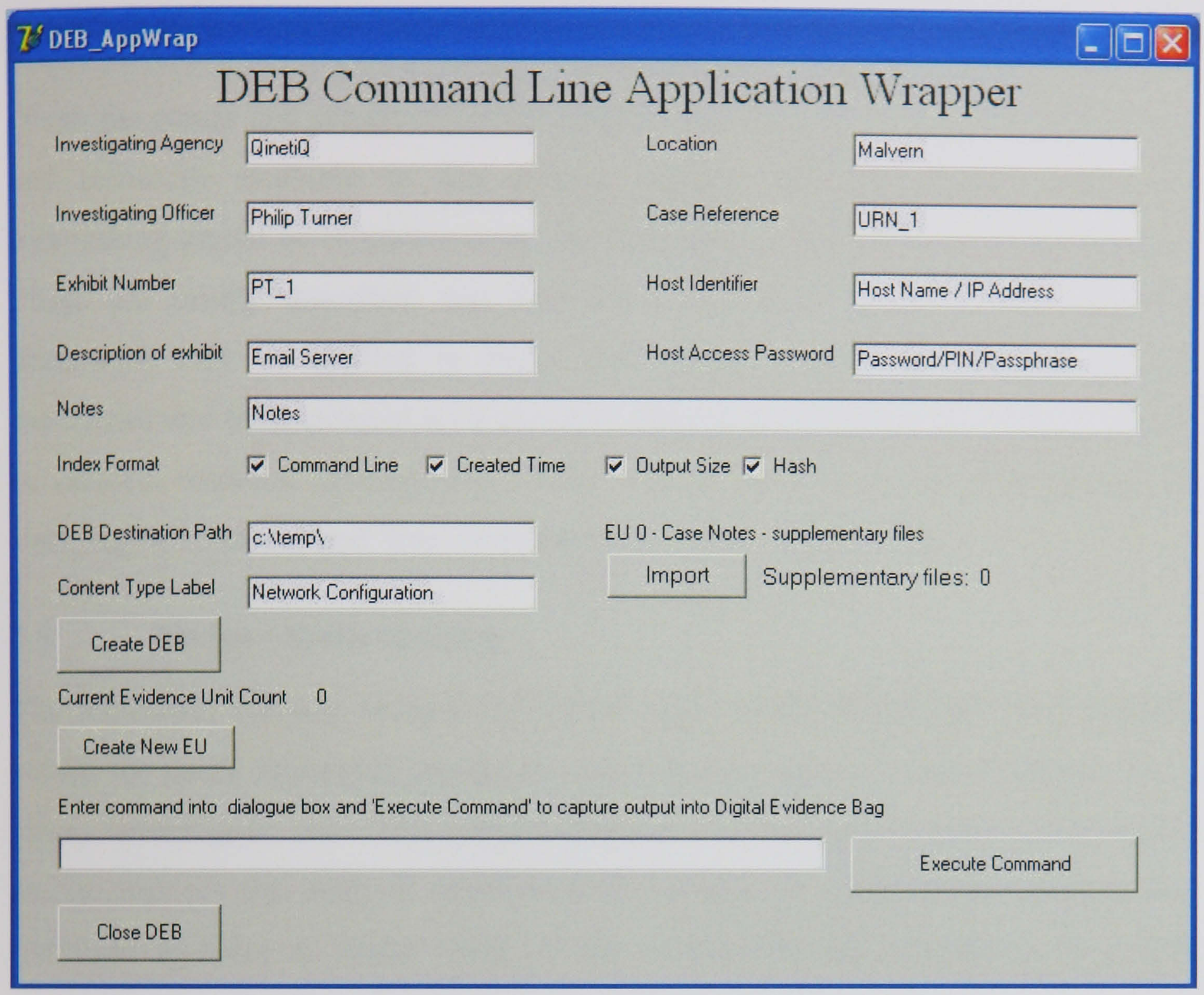

Figure 22 - DEB Command Line Application Wrapper screenshot

5 Digital Investigation Scenarios & Case Studies

Given the nature and sensitivity of the material encountered in digital investigations, and resources available to this project together with the authors experience undertaking digital investigation work, the following scenarios have been considered. These are briefly described and then consideration is given to how the DEB framework may be applied to them. The scenarios are derived from personal experience and from external texts aimed at educating and testing the skills of would-be incident response investigators. They broadly divide into two main groups i.e. static digital forensics and 'live' or dynamic environment forensics.

5.1 Device / Media Imaging

The traditional forensic scenario is that of static media acquisition. This typically entails the entire sequential capture of data from an item of media or device into a single homologous unit. This process permits a clone (exact copy) to be created and/or requires the analysis application to be able to interpret the internal data structures in order to 'make sense' of the information and represent it in a more conventional and easily interpretable way to the investigator.

An example of this would be the data capture (imaging) of a hard disk extracted from a computer. A clone may be created by writing the image in a sequential manner to another device or viewing the contents of the image through an application that can interpret the disk filing system.

The DEB framework supports this scenario by containing the media image in a single EU bag file. The corresponding EU index file holds descriptive information about the source and size of information and optionally its integrity checksum.

Additionally, the index file may contain information about the fragments (chunks) of data held in component bag files. For example, when a 250GB hard disk is captured it may be stored in files no larger than that of the capacity of a CD/DVD or other backup media.

Figure 23 shows Evidence Units containing multiple component bag files and related index entries. The size of the bag file component parts is limited to an arbitrary value depending upon that selected by the investigator.

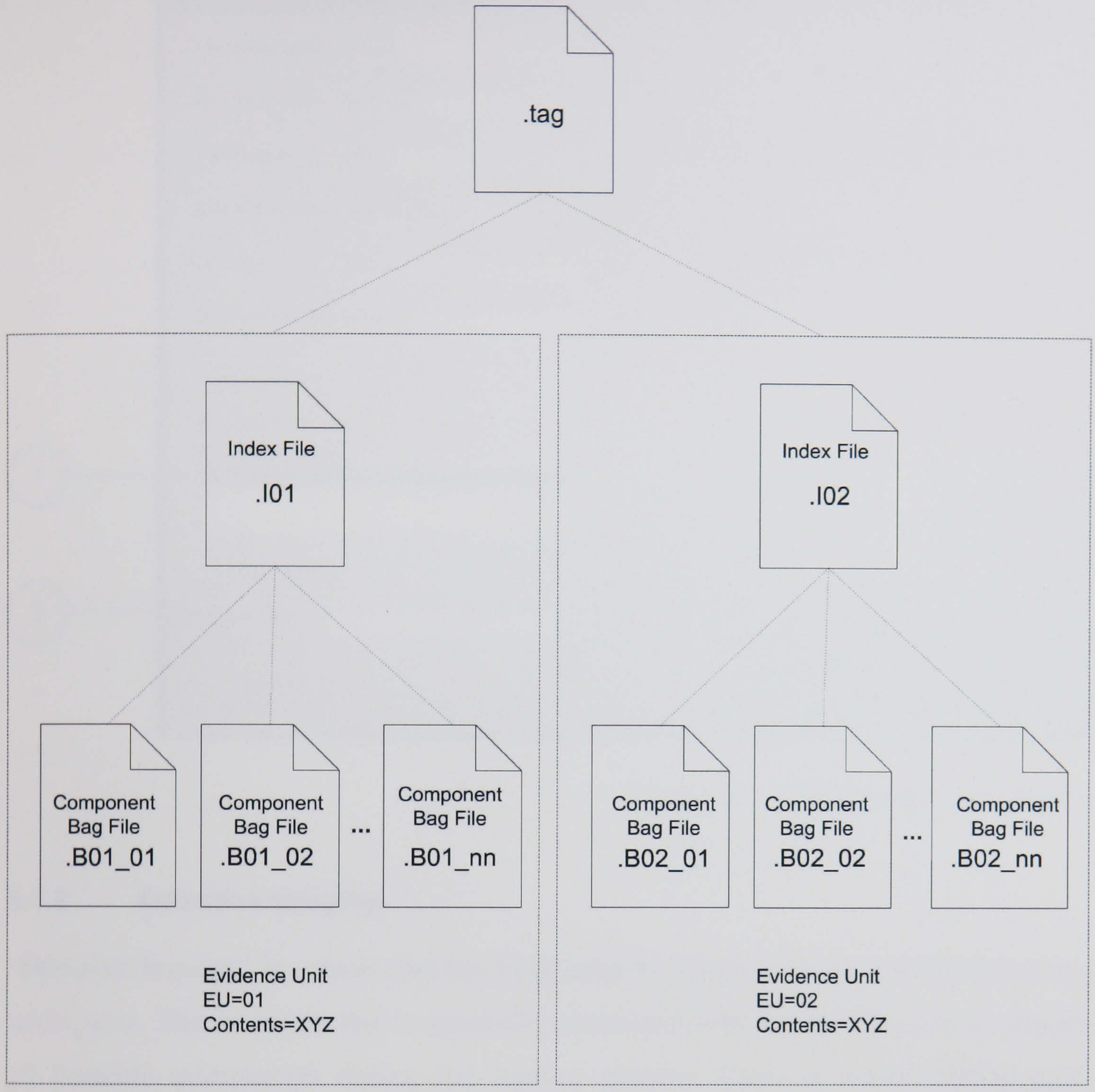

Figure 23 - DEB structure diagram

5.1.1 DEB Device Imager application

A basic media imager has been implemented to demonstrate how a DEB can store information obtained from a device in a static environment. Figure 24 shows a screen capture of the DEB Imager. This utility is based on the DEB API and allows an entire device to be imaged. The available fixed and removable devices are displayed in the dialogue box (1). When a device is selected and a DEB created, the progress of the image capture is shown in the dialogue box (2). The Index Format specified in the tag file records the Provenance Device Description <P>, the capacity of the device <Dcap>, and the time the imaging commenced <Tcre>. The Index Format utilised in this scenario is fixed and embedded in the application.

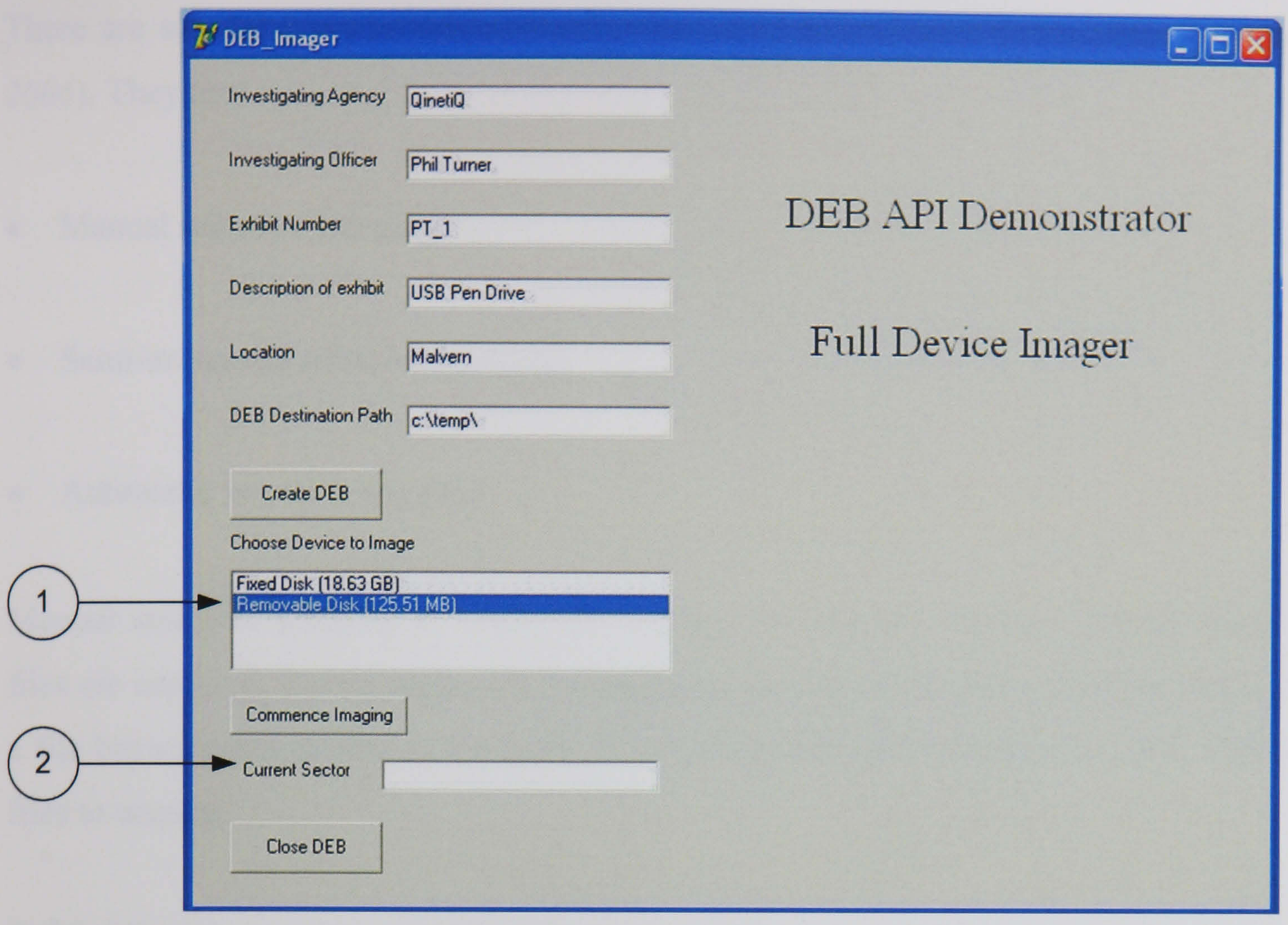

Figure 24 - DEB Imager screenshot

5.1.2 Selective Imaging

'Selective Imaging' is a term that has been used by many forensic practitioners for a while now. This is a term that is generally associated with the decision not to acquire all possible information during the capture process. Even in some official good practice guides (*ACPO 2003*) it is now recognised that '*partial or selective file copying may be considered as an alternative*' when it may not be practical to acquire everything. The usual reason for applying a selective approach is the quantity of information that may have to be acquired. Other reasons for performing a selective acquisition include, but are not limited to, forensic triage, intelligence gathering and legal requirements. There may be legal reasons why a selective approach should be adopted, for example, a case involving Legal Professional Privilege (LPP) material. Adopting a selective approach has risks associated with it as highlighted in Kenneally and Brown's papers (*Kenneally 2005a*) (*Kenneally 2005b*), but this in no way means the evidence should not be gathered in any less scientific or rigorous manner.

There are several types of selective imaging techniques that could be used (*Turner 2006*). They are:

- Manual selective imaging;
- Semi-automatic selective imaging;
- Automatic selective imaging.

Manual selective imaging is where the forensic investigator chooses exactly which files are captured. For example, the investigator can use an interface similar to that of a file browser and is able to navigate the directory tree and arbitrarily choose which files to acquire.

Semi-automatic selective imaging is where the forensic investigator decides which file types or categories of information to capture. This may be based on file extension, file signature or file hash. When using a selective approach based on file hashes it is important to record which files are present and their provenance, even though the contents of each file may not be captured. It would also be prudent to record referential hash set information.

Automatic selective imaging is where the investigator selects the source and destination devices and the imager automatically acquires the evidence. This is accomplished in a selective manner according to pre-configured parameters or the particular circumstances pertaining to the case / investigation.

The different operating modes that a selective approach presents to the investigator, combined with the flexibility and many options for classifying and grouping information, potentially makes it very complex. One of the difficulties with selective imaging is recording the provenance of each item selected. The issues associated with provenance and how DEBs store multiple provenance descriptions are discussed in section 3.4.

Additionally, the DEB framework makes provision for recording the selection criteria used in a selective approach. This information would be stored in EU0 used for case notes. For example, selecting a particular file type to capture but not actually finding any files of that type (a negative result) may be significant. The case notes could even be created in a machine processable form thereby permitting the resumption of the capture process should the first phase of the examination prove negative. This feature has not been considered in this work but it is recognised as an area for further development.

5.1.3 Intelligent Imaging

An 'Intelligent Imaging' approach is the process of capturing the knowledge and experience of domain experts and applying it to an intelligent system. This enables the investigator who is not technically proficient, and is aware only of the type of investigation they are conducting, to use this type of imager. For example, they may be investigating a fraud, intellectual property theft, or possession and distribution of indecent material, and may be unfamiliar with what file types or locations where information may reside that is pertinent to their case. They simply select the type of inquiry that is being conducted and the imager has the necessary intelligence built into it to acquire everything that would normally be relevant to the case.

There are however risks and difficulties that have to be overcome in order to adopt this approach:

- How do you go about capturing and combining the knowledge of legal domain experts and technically proficient digital domain practitioners? This is a recognised problem in the knowledge acquisition field (*Mitchell 1998*).

- How do you know that everything relevant to the case under investigation has been acquired and that evidence relating to other offences has not been overlooked?

These two points are outside the scope of this work, but need to be considered in the future when tools and systems have been developed that are capable of performing automatic selective imaging.

5.1.4 Selective & Intelligent Imaging using Digital Evidence Bags

The DEB framework is able to accommodate selective and intelligent imaging techniques by the use of flexible EU content types and Index Format specifications. This also allows a single DEB to store information obtained from a disparate range of digital devices and record multiple provenential reference types for each item of information stored in the bag file (Figure 25). The 'Category Definition' and 'Imager Configuration' file would be recorded in EU0 (case notes).

Figure 25 - Disparate digital device capture into a single DEB

The following diagram (Figure 26) shows how a DEB may be populated in a selective or intelligent data capture environment. The grouping of similar information can be seen in the particular EU content types. The grouping of information can be arbitrary but some possible grouping classifications are:

- File Extension;
- File Signature;
- File Category (e.g. Document, Pictures, Archives, Spreadsheets);
- Files within a particular hash set (e.g. paedophile image classification);
- Files not within a particular hash set (e.g. not a particular OS or application);
- System Configuration Files;
- Application Log Files;
- Files modified, accessed or created between certain times;
- Files modified, accessed or created on a particular date.

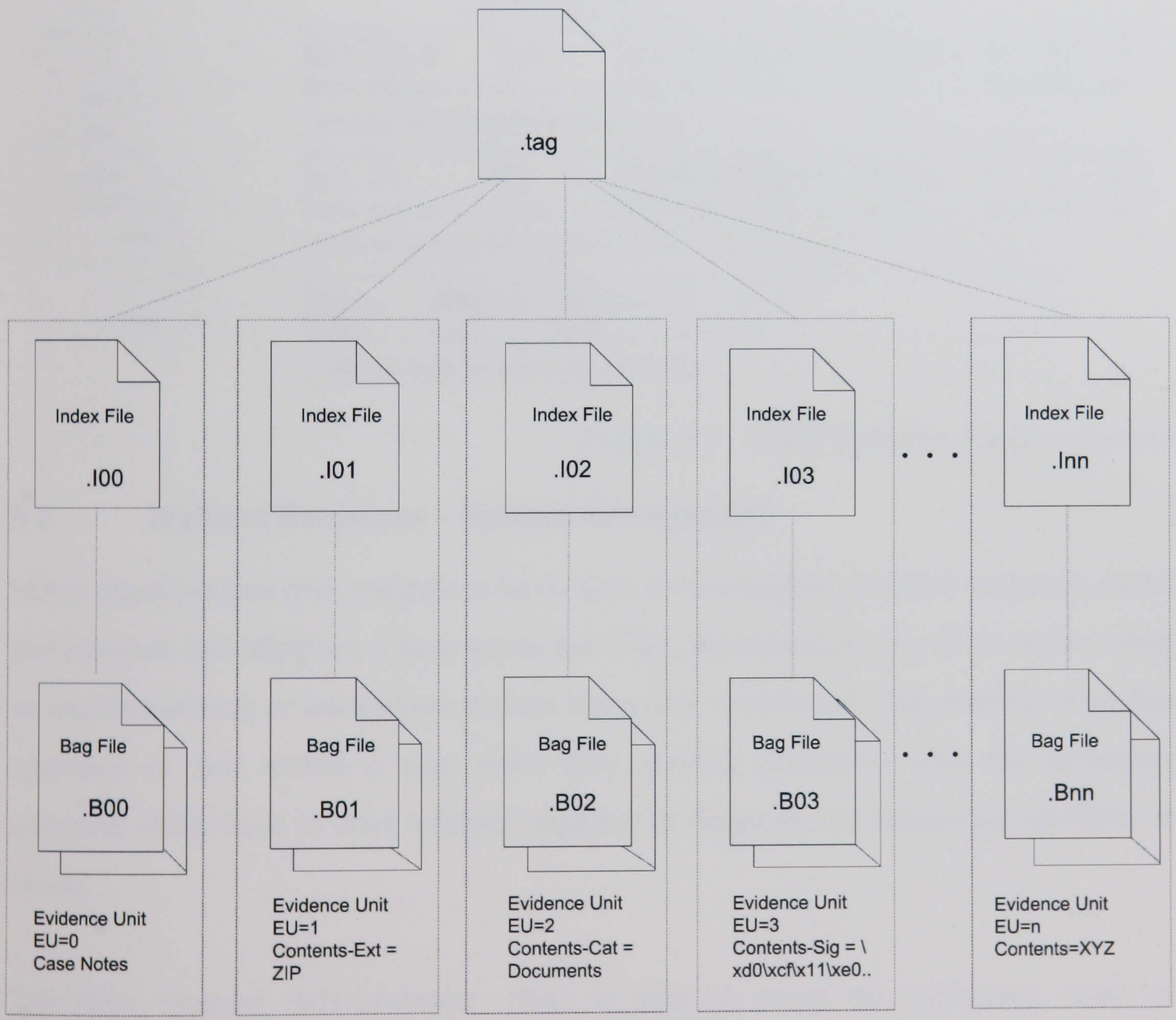

Figure 26 - DEB Structure for selective / intelligent imaging

A more detailed diagram illustrating the contents of the tag, index and bag file is shown in Figure 27. This shows how the multiple provenance storage flexibility and each component EU relates to the tag file entries.

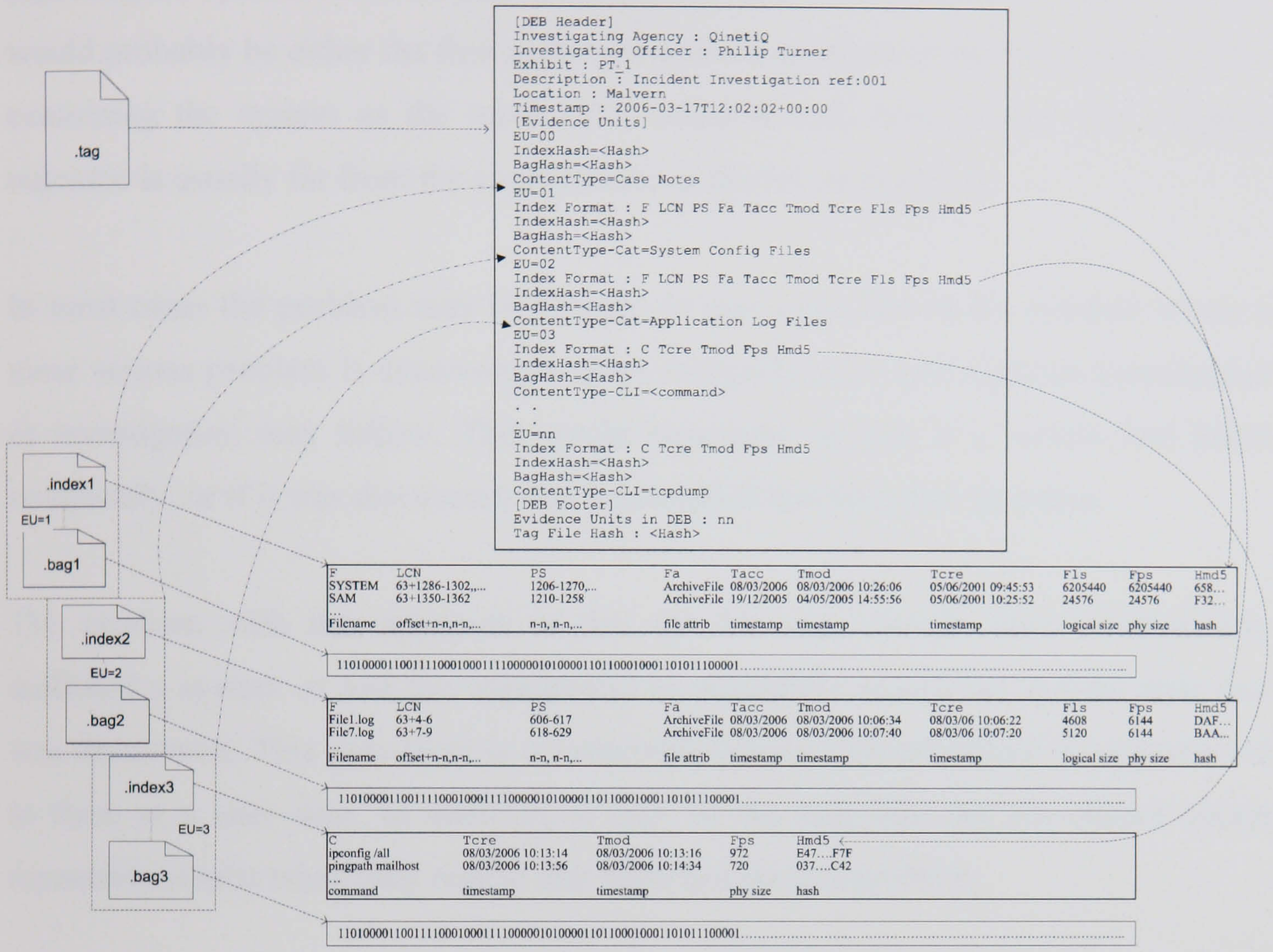

Figure 27 - DEB Selective Image structure

5.2 Incident Response – System Management

Many organisations and companies have their own computer incident response teams and network investigators. These teams are often assembled at very short notice when an incident occurs or anomalous system behaviour is detected. The problem with this approach is that within a very short time period, personnel with the necessary technical skills have to start working together to diagnose the behaviour, problem or attack .

The term 'System Administrator' (SA) is used to cover the processes, tasks or operations performed by the system administrator, system operator, incident investigator, security administrator or network investigator. In effect, anyone who may have any role in determining the problem, cause or effect of any abnormal or unusual system behaviour.

At the onset of the incident all that may be known is that something on a system or network is not working as expected and the goal is usually to restore the service, capability or system to its normal level (*Turner 2007b*). In the first instance the SA would probably be either the first person to detect a problem or the first person to start examining the system as the result of a helpdesk call from a user. The eventual outcome is usually far from the expectations of the SA or examiner.

In most cases the problem may be benign, or user error, but in the instance where a more serious problem is discovered a more comprehensive and rigorous examination or investigation may follow. This would especially be true if a system had failed completely, or if it was discovered that information had been lost or stolen.

The problem with this approach is that the SA may already have inadvertently modified a system or lost the opportunity to accurately record the system state that was discovered. This puts them in a vulnerable position should blame be apportioned to them at a later date. In addition, it may be the case that the SA cannot obtain repeatable results when they realise that there is a serious problem.

Ideally a SA should be forensically trained and thereby be part of a coherent forensic approach with the incident response or network investigation procedure that they might normally adopt. The forensic aspect should be an integral part of any system administration role that is responsible for operating any business critical or business operational system and should be part of a forensic readiness strategy (*Rowlinson 2004*).

5.2.1 Incident Response - Common Tools

The SA usually has a vast range of tools available at their disposal to monitor system performance, determine system configuration or to fault find system problems. The majority of these tools and utilities are very specialised in the function they perform and the information they provide. Many of these tools are console applications and run as command line utilities. Furthermore, they are often packaged with the operating system.

The problem with these tools is they are designed to provide information, but are not designed to provide any form of integrity assurance or record when those utilities were executed. From a forensic perspective they provide no audit record about the timestamp or actions taken, or results returned from running those utilities.

Admittedly the SA, with forethought, could choose to pipe the output to a log file, but this still has no mechanism to assure the integrity of the output data. Furthermore, the SA would rarely keep handwritten notes of the actions taken, or results obtained whilst trying to diagnose and examine a system.

This typical scenario results in the SA being unable to justify or even demonstrate to colleagues the system state as they found it, never mind being able to assure that they were not the cause or a contributor to the problem or incident.

There are many tools that may be used to diagnose system behaviour (*Sorenson 2003a*), (*Sorenson 2003b*), (*Carvey 2001*), (*Microsoft 2007d*). In addition to these tools there are toolkits available with many of these applications already compiled into a compendium (*Helix2007*), (*Foundstone 2007*).

This situation is further endorsed by the 'Hacker's Challenge' (*Schiffman 2001*) which sites 20 scenarios where by systems were compromised and investigations started to identify the cause of the problem. An analysis of these scenarios showed that the sources of information used to base an investigation came from two main areas; log files and command line utilities.

5.2.2 System and Application operation log files

The operating system log files and application log files are excellent sources of information to establish system behaviour and problems. These are typically used to record a diverse range of actions such as user logon and logoff, system service start up and print job auditing. Other more obscure tasks can also be recorded in log files, such as when anti-virus software commenced or when Microsoft Office documents were opened and created (*Microsoft 2007e*).

In (*Schiffman 2001*) the following system logs were used as sources of valuable hacker information:

- Microsoft IIS server logs (*Microsoft 2007f*);
- MS Exchange server logs (*Microsoft 2007g*);
- Virtual Private Network (VPN) logs;
- Network Intrusion Detection System (NIDS) logs, snort logs;
- Syslog;
- Firewall logs;
- Router logs;
- Physical access logs.

The main feature most system and application logs have in common is that they record the date and time stamp of when a particular event or action took place. From an investigation perspective (although this is very useful and important information) it lacks any form of integrity check or error detection information.

The correlation of log information with other information found on the system or other connected systems is vitally important if the log information is to be relied upon. This is because some log formats are easily tampered with and could involve entries being deleted, modified or timestamps changed. The detection of these alterations can be difficult if the log file information is considered in isolation. The correlation of information and timestamps is discussed further in (*Forte 2004)* and (*Schatz 2006)*.

Wrapping log files in a DEB addresses these issues by providing integrity, provenance and continuity mechanisms. The investigator is also certifying the state of the log file at the time of capture there by allowing subsequent changes to be identified whether due to accidental or malicious means.

5.2.3 System Administration Tools - Command line utilities

All operating systems come with some utilities that allow users to create and manage information, with the potential to connect to other systems and transfer information between those systems. This basic functionality and the tools that provide such functionality can also be used by system administrators during the course of an investigation to both establish the operational state and connectivity of a system.

The following lists the common utilities (*Schiffman 2001*) used on Unix systems to obtain system information:

- ps -Af - list all processes running on a system;
- netstat - to capture current state of the network connections;
- lsof - used to find out which processes had files open;
- telnet - used to connect to remote host or open port on a system;
- date - used to print or set the system date and time;
- ls - list directory contents;
- ping - send ICMP ECHO_REQUEST packets to network hosts to ascertain if the address exists and is responsive.

The following lists the common utilities used on a Microsoft Windows OS to obtain system information:

- Fport – used to identify unknown open ports and their associated applications, http://www.foundstone.com/us/resources/proddesc/fport.htm [Accessed 10 September 2007];

- Attacker – TCP/UDP port listener, http://www.foundstone.com/us/resources/proddesc/attacker.htm [Accessed 10 September 2007];

- Netstat - Displays active TCP connections, ports on which the computer is listening, Ethernet statistics, the IP routing table, IPv4 statistics (for the IP,

ICMP, TCP, and UDP protocols), and IPv6 statistics (for the IPv6, ICMPv6, TCP over IPv6, and UDP over IPv6 protocols),

http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/netstat.mspx [Accessed 10 September 2007];

- Pslist – used to show process and thread lists on local and remote systems, http://www.microsoft.com/technet/sysinternals/Utilities/PsList.mspx [Accessed 10 September 2007];

- Auditpol – list systems auditing policy – Windows Resource Kit;

- Psloggedon – displays logged on users, http://www.microsoft.com/technet/sysinternals/Utilities/PsLoggedOn.mspx [Accessed 10 September 2007];

- Dumpel – dump event log for local or remote system, http://www.microsoft.com/windows2000/techinfo/reskit/tools/existing/dumpel-o.asp [Accessed 10 September 2007];

- Regdump – tool for outputting all or part of the registry.

The main feature that all these utilities perform is to provide information for the SA about the system state, or to perform a specific function. In contrast to the system and application logs they do not record the date and timestamp of when the particular action took place.

From a digital forensic investigation perspective, although these command line utilities and various log files are very useful, they lack any form of integrity check or error detection information attached to their output. To address this problem the DEB framework can not only be used to encapsulate the output generated by these commands but also to record timestamp and integrity information. This is accomplished using the DEB Command Line wrapper application (see section 4.3).

5.3 Magnetic Stripe Card Cloning devices

The DEB framework permits the storage of digital information acquired from a range of disparate digital devices (*Masters & Turner 2007*). For example, information obtained from magnetic stripe card cloning devices can be stored in a DEB.

A magnetic stripe card (swipe card) is a type of card capable of storing digital data by recording a magnetic pattern within a stripe on the reverse of the card. Swipe cards are commonly used for applications including credit cards, department store (loyalty) cards and mobile (or cell) telephone 'top-up' cards. There are three data tracks within a magnetic stripe. A credit card typically uses only tracks one and two.

Track 1 typically stores the primary account number, card holder's name and card expiration date. This track can also be used to contain 'discretionary' data maintained by the card issuer. Depending on the issuing authority, this discretionary data may, but not necessarily, be used for PIN or card verification and transaction counting purposes.

- Track 1 – 76 alphanumeric characters
  - Start Sentinel = %
  - Format Code, B = Bank/financial format
  - Primary Account Number (PAN), up to 19 digits
  - Name, 2-26 characters
  - Expiry Date

  Example

  %B0123456789123456^MR A SMITH^0612…?

Track 2, developed by the banking industry, typically stores a copy of track 1, but without the cardholder's name, and a 'service code' entry related to card security functions, such as the type of transaction permitted (cash only, goods and services only or ATM (Automatic Teller Machine – 'hole-in-the-wall') with PIN verification).

- Track 2 – 37 numeric characters
  - Start Sentinel = ;
  - Primary Account Number (PAN), up to 19 digits
  - Expiry Date – 4 characters
  - Service Code – 3 characters (sss)
  - Discretionary Data (DD) - PIN / Card Verification

Example

; 0123456789123456=0612sssDD…?

Track 3 - Not usually used for financial transaction cards

- Track 3 - 104 numeric data characters
  - Start Sentinel = +
  - Field Code (FC)
  - Primary Account Number (PAN), up to 19 digits

Example

+ FC0123456789123456=…?

A number of ISO/IEC standards define both the physical layout and construction details of a magnetic swipe card and the data format of the tracks:

- *ISO/IEC 7810* – Physical characteristics of credit cards;
- *ISO/IEC 7811* (1-6) – Embossing, Track location, Lo / Hi coercivity;
- *ISO/IEC 7813* – Financial Transaction Cards;
- *ISO/IEC 4909* – Card Data Format – Track 3.

There are many magnetic stripe card readers or skimmers, as they are commonly known, produced by a small number of manufacturers. These devices are marketed for legitimate commercial retailers. They also have become increasingly used for fraudulent activities. An example of a portable magnetic stripe device is shown below (Figure 28 and Figure 29) with a brief synopsis of their features:

- Standalone, battery powered – CR2032 button cell
- Size - L50mm x W30mm x H38 mm
- Reads 3 Tracks
- 512K bytes memory – up to 2048 records
- RS232 / USB interface connections
- PIN protected – 4 digit
- Software deletes records/wipes information from device when saved

Figure 28 - MSR-500M (Mini –123) Magnetic Swipe Card Reader

- Ability to read and write magnetic track data
- Track reading/writing options 1,2&3, 1&2, 2
- Hi / Lo Coercivity
- Serial / USB / PS/2 Connection types
- Can be used to clone magnetic stripe cards

Figure 29 - MSR206 Magnetic Swipe Encoder

The following shows a sample of the data that would be extracted from a Mini-123 card reader. The card details are contained in records containing concatenated track information:

Unit Login ID : 0000

Actual Date & Time : 13/10/06 12:12:30

Unit Date & Time : 200703142328394

Product Version : 1.0R16

Number of records : 0001

000,;8944129990123456789=99121010000000000?2006/11/09 09:35:39 53F

This track information requires interpretation in order to present it in a more human readable format. The decoded track information from the Mini-123 is shown below:

Record: 000

Timestamp 09:35:39 09/11/2006

No Track 1 data

No Track 3 data

TRACK 2

Account Code: 89441299901234567 89

Valid From: 12/99

Some of the applications packaged with skimmers are almost anti-forensic in nature as they actually delete the information contained in the skimmer once downloaded to a computer. Furthermore, they may even require a basic password in order to access the stored data. To overcome some of these problems, bespoke applications are required by the forensic investigator to connect to the device and extract the information in a forensically sound manner.

The following example (Figure 30) shows how the data contained within a Mini-123 magnetic stripe card reader can be stored in a DEB. It can be seen that integrity check information can be readily associated with the record information extracted from the device.

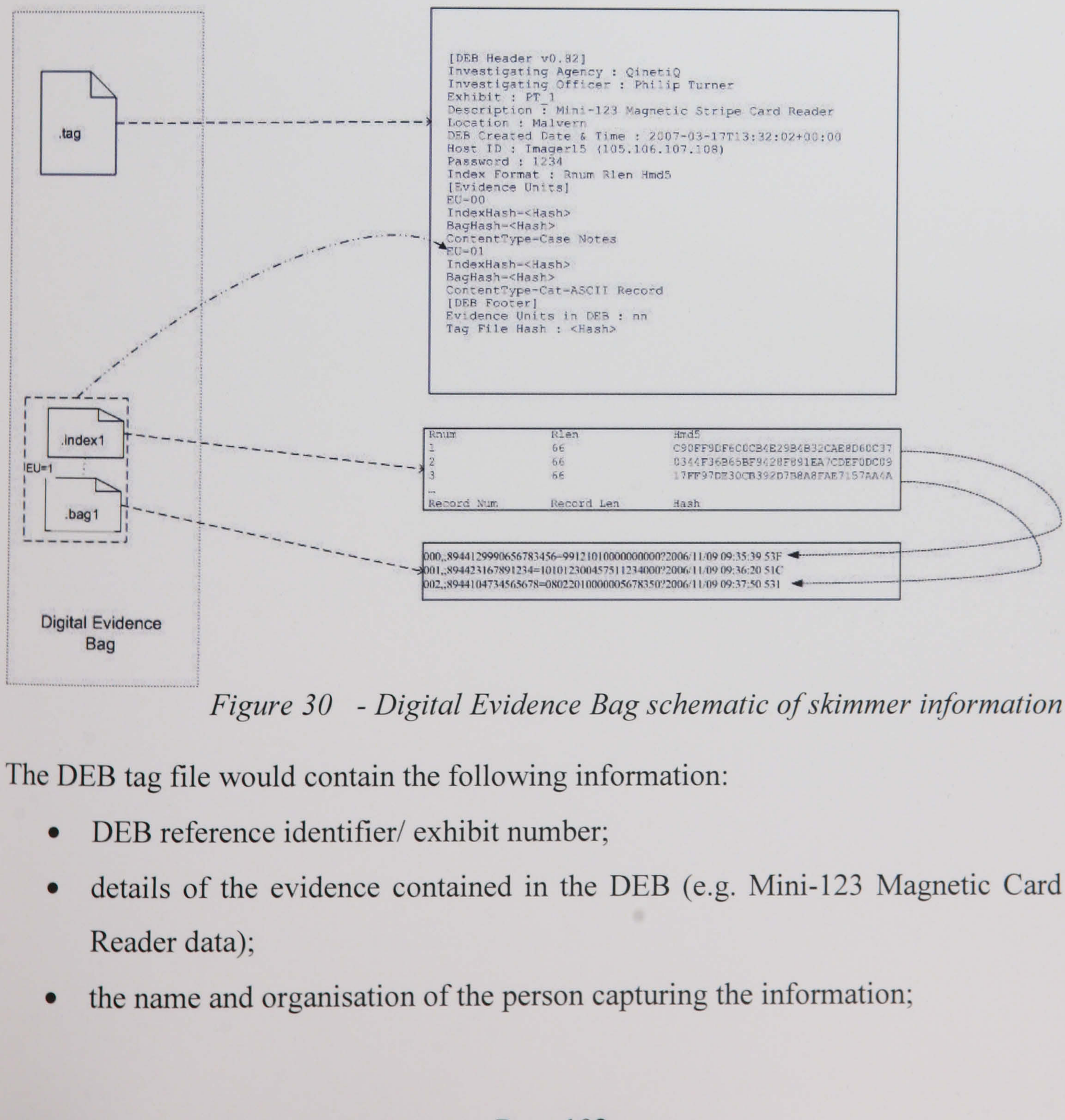

Figure 30 - Digital Evidence Bag schematic of skimmer information

The DEB tag file would contain the following information:

- DEB reference identifier/ exhibit number;
- details of the evidence contained in the DEB (e.g. Mini-123 Magnetic Card Reader data);
- the name and organisation of the person capturing the information;

- the date and time the capture process started;
- a list of Evidence Units (EUs) contained in the DEB;
- a hash of each Index and Bag file contained in the DEB;
- tag seal number comprised of a hash of the tag file to date; this is equivalent to the traditional seal number;
- the format definition of information stored in the Index file.

The index file in this scenario utilises a basic index reference structure (Record Number <Rnum> and Record Length <Rlen>) and associated integrity check information (MD5 hash <hmd5>) associated with the actual magnetic swipe card data extracted from the Mini-123 device that is contained in the bag file.

6 Comparison of DEB framework with other Evidence Storage Formats

The DEB framework is now compared with a number of other formats, both EnCase and Raw have been discussed in Chapter 2 as they are the most prevalent and supported formats currently in use. Details of the other formats discussed in this chapter are available as detailed below:

6.1 Digital Forensic Image Storage Formats

6.1.1 Advanced Forensic Format (AFF)

AFF is developed by Basis Technology Corporation and Simson Garfinkel, and implements a segmented file specification. Segments can include media image data or associated metadata. The format provides an extensible approach to the storage of metadata through the use of name/value pairs. The format also supports image data compression through the use of ZLIB and MD5 or SHA-1 integrity assurance digests. Further details of the format together with a library of tools are available from www.afflib.org [Accessed : 29 September 2007].

6.1.2 Expert Witness – EnCase - SMART

The Expert Witness format is the predecessor to that currently used in the EnCase and SMART toolsets. It was used for the storage of disk images. The format is not published but is believed to be virtually identical to the published Expert Witness Compression format used by SMART (see below). Due to a legal dispute between the creators of this format it is worth mentioning because of its historical interest in forming the basis of the EnCase format.

6.1.3 EnCase

The current EnCase format is an enhancement of the earlier Expert Witness format. Its basic structure and the metadata that it is capable of storing is described in earlier chapters but its detailed specification is not published. It is capable of storing disk images and those acquired from Palm PDA devices. It utilises MD5 digests to assure

the integrity of the captured media data and a series of Adler32 CRCs for error detection across smaller blocks of data (usually 32K).

6.1.4 SMART

This is defined by ASR Data and is also used for hard disk images. There are actually two SMART formats. The default format stores metadata in separate text files. Although the layout and structure of this is not published, the contents may be easily viewed. The second format produced by the SMART toolset is the Expert Witness Compression format. It stores the same set of metadata as the EnCase format. Specification details are available from www.asrdata.com/SMART/whitepaper.html [Accessed: 29 September 2007].

6.1.5 Generic Forensic Zip (Gfzip)

The Generic Forensic Zip format is an open design produced by Rob J. Meijer. It uses data structures similar to AFF. However, the storage of media information and associated metadata is different. The format stores metadata after the acquired media data within the file. The format stores disk images that can be uncompressed (that are compatible with 'raw' format images) or compressed. The SHA-256 digest algorithm is used for integrity assurance. Additionally, X509 certificates and signatures are used to authenticate the contents of the image file. Other features of this format include the support for a 'packed' storage method which reduces the storage required for identical blocks of information and support for encryption. Details of Gfzip are available from www.nongnu.org/gfzip/filespec.html [Accessed : 29 September 2007].

6.1.6 ProDiscover

This format is defined by Technology Pathways for use in their range of products and is used for hard disk images. The format incorporates metadata in the image data header structure and supports the use of MD5 or SHA-1 integrity digests. The metadata that the format is capable of storing includes:

Name of Technician;

Name of Source Disk;

Make of Hard Disk;

Number of free sectors, Bad sectors and clusters;

Number of image splits;

Current Timezone and whether daylight saving is in operation.

Compression is also supported by dividing the source into 1MB blocks although no algorithm is given in the specification. The format is open and published and is available from www.techpathways.com/uploads/ProDiscoverImageFileFormatv4.pdf [Accessed : 29 September 2007].

6.2 Format Comparison

From the outset, the DEB framework was designed to be easy to understand and to provide a more flexible and efficient way of storing digital evidence to meet the requirements identified in section 2.4. This is achieved by imitating the terminology and features used on the physical evidence storage containers of bags, tags and tamper-proof seals (section 3.1.1) that are used universally by law enforcement agencies. However, this concept is not the case with all other formats used for evidence storage.

DEBs are capable of recording descriptive information and metadata relating to the source of the information contained within it. For example, the make, model, serial number, photographs and descriptive information of the device being captured. Other formats such as the EnCase Evidence file format and ProDiscover formats are limited to the storage of a short textual description, but no other metadata can be associated with the evidence at commencement of the capture process.

Additionally, DEBs can be used to record the provenance of information contained within them. For example, the folder path of a particular file, its physical sector location and the logical cluster number can all be stored. In contrast, other evidence storage formats capture information into a single data block and rely upon the analysis tools to decode and interpret the inherent filing system structure.

Because of the volume of information stored on modern storage media the acquisition of data into a single file would make backup and duplication on other media cumbersome. To alleviate this problem, and in common with other storage formats,

DEBs permit the fragmentation of information into segments ('splits' in ProDiscover terminology), the size of which can be specified by the investigator.

One of the major differentiators between the DEB approach and every other storage format is that it permits the storage of selected information from the source media. For example, if a selective imager is used, particular categories of information can be captured into discrete DEBs or EUs together with its provenance. This facility is not supported in any other format. The closest feature that is available is the logical evidence file that can be created using EnCase. This permits the extraction of particular data from a case into a new evidence file but does not store any associated provenance information.

The DEB framework uses an index file to record the type and source of information contained within the bag file. This feature is also not available with any other evidential storage format. Dictionary indexes are used by some forensic analysis tools to permit a fast keyword search capability but are not used for the storage of information. The dictionary indexes are created following media acquisition.

In common with other digital evidence storage formats the DEB framework uses cryptographic hash signatures (digests) to assure the integrity of the information contained within the file. However, whereas the hash functions used within other formats are fixed (e.g. MD5 digest), the DEB framework is extensible. The initial DEB definition permits the use of MD5 or SHA digests to be specified, additional algorithms can be 'plugged in' as they are developed in the future without requiring wholesale format redefinition.

In addition to assuring the integrity of any data captured in a bag file the DEB format also uses identical integrity assurance mechanisms for the case metadata stored in the tag file. This is in contrast to the EnCase and Expert Witness formats which rely solely on weaker CRC integrity algorithms for this category of data.

The DEB format also allows for flexible data storage from any type of digital device. Only the 'raw' data format permits this whereas the AFF, EnCase, Gfzip and ProDiscover style formats are limited to digital media storage (hard disk, CD, DVD)

and encapsulation. DEBs achieve this flexibility by using a series of meta-tags to specify the index file format and structure. This system is also extensible, so should a new category of device information be required to be stored in a DEB, then this can be accommodated with a minor revision to the DEB definition. The meta-tags defined currently permit DEBs to store hard disk images, logical files, command line output, network traffic, text-based records, memory dumps and other digital information.

The DEB format also permits the multiple provenance locations of information to be captured. For example, for a given file the folder path, logical cluster number within the volume and physical sector number locations on the media can be stored. This feature is also not supported in any other evidence storage format.

In contrast to all other digital evidence storage formats, the DEB framework contains a dynamic and a static component. The tag file provides a mechanism to record all applications and processes that are used to access the whole DEB or individual EUs. Other storage formats are purely static containers; such that once digital data has been captured, no aspect of the container may be changed without rendering either the whole of the evidence or a small proportion of it unreliable. The DEB framework also permits the DEB to be 're-sealed' on completion of an examination or analysis operation. The ability to record that the DEB has been accessed together with a signature, functional description and timestamp of the operation being performed is also unique.

Some digital evidence storage formats permit the captured data to be compressed. The algorithm used in all current formats is fixed. Additionally, the compression function would be applied over the whole of the captured data stream. In some cases this is not efficient since trying to compress already compressed data can be time consuming and result in larger generated output than the original. In contrast to this the DEB framework is capable of supporting different compression algorithms and optionally certain types of information could be selected for compression. For example, it may be desirable to capture word processing documents with compression into one EU, and use another EU to store uncompressed movie files.

The DEB framework also supports the use of encryption at the granularity level of EUs. This allows differing levels of protection to be placed around certain categories of information.

Table 3 compares the DEB framework with the other commercial tools described earlier; this is based on the authors knowledge and experience in this field. Additionally, the features of the tools are correlated with the requirements of imaging and analysis tools identified in section 2.4 and Table 1.

The following table summarises the features provided in the DEB framework when compared to other common digital evidence storage formats and correlated to the requirements identified in section 2.4.

Feature	DEB	AFF	Expert Witness / EnCase / SMART	Gfzip	Pro Discover	Raw	Requirement
Store source metadata	✓	✓*1	✓*1	✓*1	✓*1	×	Provenance
Store multiple provenance description information	✓	×	×	×	×	×	Provenance
Fragmentation of evidence into multiple file	✓	✓	✓	✓	✓	✓*2	Efficiency
Permits selective information capture	✓	×	×	×	×	×	Proportionate
Catalogue or index of stored information	✓	×	×	×	×	×	Repeatability / Reliability
Compression support	✓	✓	✓	✓	✓	✓*2	Efficiency
Integrity assurance mechanism	✓	✓*4	✓*5	✓*6	✓*4	✓*2	Integrity
Encryption support	✓	×	×	✓	×	✓*2	Integrity
Permits selection of integrity assurance algorithm used	✓	×	×	×	×	✓*2	Integrity
Permits selection of compression algorithm used	✓	×	×	×	×	✓*2	Efficiency
Permits selection of encryption algorithm used	✓	×	×	×	×	✓*2	Integrity
Supports static digital media capture (HDD/CD/DVD)	✓	✓	✓	✓	✓	✓*3	Disparate Devices
Supports dynamic / live information capture	✓	×	×	✓	✓	✓*3	Disparate Dev. / Environments
Records access to evidence - dynamic component	✓	×	×	×	×	×	Audit Trail
Capable of recording process and analysis applications used to access evidence	✓	×	×	×	×	×	Continuity

Table 3 - Format Comparison Summary

*1 limited to small amounts of textual information
*2 - through the use of other utilities. E.g. 'split' to fragment a file, 'md5sum' to generate an integrity digest
*3 any block device supported by 'dd'
*4 - MD5 or SHA-1 digests
*5 - MD5 digest only
*6 SHA-256 digest only

6.3 DEB Associated Work

The term 'Digital Evidence Bag' has also been used in another piece of research work that was undertaken for the Air Force Research Laboratory (AFRL) by Wetstone Technology.

The paper is entitled *SI-FI (Synthesizing Information From Forensic Investigations)*, G. Hosmer, G. Gordon, C. Siedsma, J. Hosmer - AFRL-IF-RS-TR-2002-12, February 2002. Available from:

http://stinet.dtic.mil/cgi-bin/GetTRDoc?AD=ADA402491&Location=U2&doc=GetTRDoc.pdf [Accessed: 24 September 2007].

The term DEB was used within the paper for the definition of an XML schema that describes all the various items of information that could be stored as part of a digital investigation. The paper does not define any specific format for digital evidence storage.

This topic was re-visited in the paper entitled *Digital Evidence Bag*, C. Hosmer, Communications of the ACM, February 2006, Volume 49, No. 2, Pages 69-70. This paper refers to the SI-FI project and highlights the need for auditing in the digital world and the need to maintain permanent audit records throughout the lifecycle of a system, although no further technical detail is provided. Apart from these two publications the work on the SI-FI project appears to have ceased with no further publications or technical details available.

The requirements of the SI-FI project are:

1) *The SI-FI system must support a variety of evidence collection, extraction, examination and analysis technologies;*
2) *The SI-FI system must be heterogeneous in order to support evidence collection, examination and analysis technologies on multiple heterogeneous computing platforms;*
3) *The approach must support a broad range of cyber-forensic data;*

4) *Digital evidence by its very nature can be collected globally; therefore, synthesis of collected evidence must be possible regardless where the physical systems or networks exist;*
5) *The examination or sharing of the collected evidence must also be globally accessible by experts, investigators, and examiners anywhere and at anytime;*
6) *The integrity of digital evidence must be maintained. The SI-FI system must protect the integrity of evidence throughout its useful life;*
7) *The privacy of the digital evidence especially when communicated over public networks such as the internet must be maintained at all times;*
8) *Access to digital evidence must be regulated by strict policy and only those with authenticated privileges should be granted access to specific evidence;*
9) *The digital evidence must be tamperproof;*
10) *Any SI-FI system must be accessible and useable by a variety of users with different levels of technical knowledge. Whenever possible the handling, policies and procedures applied to digital evidence should mimic established evidence handling processes;*
11) *All derived evidentiary data should be referenced to the original source evidence;*
12) *Detailed audit trail information must be maintained regarding each piece of evidence collected. This can include information pertaining to how evidence was collected, by whom, when, how (what tools were used), investigative techniques employed, etc.*

The SI-FI project uses three XML schema representations, these are CASE, Digital Evidence Descriptor (DED) and Digital Evidence Bag (DEB).

CASE

The grouping of digital evidence at the highest level is accomplished by defining a Case (CASE) XML schema. A CASE contains information such as the identity of the person responsible for the case, legal information, and references (URLs) to all of the DEB(s) associated with the case. There is no cryptographic binding by the DED of evidence to a specific case.

DED

Digital evidence is uniquely identified and preserved by binding information (such as who collected it, what was collected, why it was collected, where it was collected, and how it was collected) cryptographically with the evidence file. A Digital Evidence Descriptor (DED) XML schema was defined to facilitate storing information and its cryptographic binding to the evidence file.

DEB

The Digital Evidence Bag (DEB) XML scheme provides a container for defining the location of a Digital Evidence Descriptor (DED) and the location of the digital evidence file to which it is cryptographically bound. Digital Evidence can exist in multiple physical locations, yet still be the same piece of evidence. If the physical location of the evidence file and the actual evidence file were bound cryptographically the evidence would be bound to exist in a specific location. Organisations can share digital evidence by transferring DED(s)/evidence file(s) and store the evidence in new DEB(s).

The SI-FI project uses a web based Client-Server architecture to support these schemes. However specific design details or prototype demonstrators were not available to provide a direct comparison with the DEB framework enumerated in this thesis.

As expected there is a fair degree of overlap between the SI-FI requirements and the Digital Evidence Bag framework and requirements described in Chapters 2 to 4. However the SI-FI project does not identify the following core requirements:

- Repeatability;
- Proportionality;
- Efficiency;
- Tool certification;
- Training / education aid.

7 Conclusions & Recommendations for Future Work

7.1 Conclusions

The objectives of this work (defined in section 1.2) was to produce an efficient scalable framework that is capable of recording the provenance, process, methods, tools and actions taken by the investigator. This is achieved by implementing and enhancing the methods used in traditional physical evidence handling and applying them in the digital environment. The DEB framework provides a more comprehensive provenance, continuity and process recording and audit mechanism than the tag used in traditional evidence handling methods. Additionally it applies stronger integrity assurance mechanisms than tamper-proof bags and seals. Information can be further protected using encryption methods or more efficiently use available storage space using compression.

The DEB framework defined in Chapter 3, provides a totally new method and facility for the storage of digital evidence. It uses terminology to describe the various components of the framework that are analogous to those already used in the law enforcement, legal and investigative domains. The simple terminology makes it easy to comprehend the function or purpose of each component. This is essential for allowing it to be easily understood by the judiciary and members of the public who may be unfamiliar with digital evidence concepts. Therefore it does not obfuscate or detract from the evidence that is stored within a DEB.

The framework permits new capture methodologies (section 5.1.2 and 5.1.3) to be developed which provide viable alternatives to the 'capture everything' approach that has been used extensively in the digital forensic arena until now. The adoption of this type of flexible approach should permit the tools and techniques that are used to acquire, process and analyse digital evidence to further develop and also to better keep pace with the ever increasing quantities of digitally stored information.

Digital forensic techniques have been severely inhibited by the lack of standardisation in the field. Most tool vendors develop their own format (section 2.2.1) which is often not published thus preventing independent verification, assessment or compatible

tools to be developed. This 'black box' approach to evidence storage is short-sighted and stifles the development of new analysis tools. It effectively 'locks in' an investigator to a specific toolset or requires the conversion into another format thus resulting in additional overhead in terms of time, storage and labour cost.

The presentation of the DEB framework to the International digital forensic community has created much interest and re-evaluation of the methods used to capture digital information for evidential purposes. The creation of the Common Digital Evidence Storage Format working group (*CDESF 2006*) is recognition of the value in taking a fresh look at the basic building blocks that form the cornerstone of a digital forensic investigation – the capture methods used for digital evidence. Also highlighted by the difficulties encountered in progressing the working group activity is acknowledgment that it is a non-trivial task that is being undertaken and that considerable time, effort and foresight are required to conclude this work. The easy option is to maintain the status quo and utilise whatever facilities any individual tool vendor chooses to implement. The creation of a formally established International Standards activity would improve:

- digital forensic tool verification;
- practitioner training;
- laboratory accreditation;
- inter-jurisdiction investigations and evidence transfer;
- academic understanding of digital forensic problems and issues;
- legislation.

Additionally, the novel approach taken by the DEB framework is apparent with the patent application documentation that has been prepared during the course of this work (Appendix C). To date the main concepts and claims within this approach have not been countered by any national or international searches for prior art conducted independently by the UK Patent Office or associated International counterparts.

This thesis has defined and illustrated the DEB framework and applied its concepts in a number of scenarios (Chapter 5) that are commonly encountered in digital forensics.

It demonstrates that a common structure can be applied to the storage of information acquired in traditional static and dynamic live incident response environments. A DEB API (Chapter 3) has been defined that provides the functionality required of applications that create and analyse the information stored within a DEB. A Delphi Pascal API implementation has been developed (Chapter 4) that has been incorporated into a number of application demonstrators that successfully illustrate the flexibility of the DEB framework.

Table 4 summarises the digital forensic imaging and analysis tool requirements identified in section 2.4 and correlates these with features provided by the DEB framework.

Requirement	DEB framework features
Record provenance	Stored in DEB Header – see sections 3.3.1.1, 3.4.1 and 4.1.
Record continuity	Stored in TCBs – see sections 3.3.1.4. 3.4.5 and 4.2.
Maintain integrity	Achieved through the use of cryptographic hashes in DEB tag and index components – see section 3.4.3.
Record audit trail of process / assumptions / through processes / application used / actions / functions / errors.	Stored in TCB and Case Notes in EU0 – see sections 3.3.1.4, 3.4.6, 4.3 and 5.2.
Repeatability / reliability	Audit trail of application version and functional description recorded in TCB – see section 3.3.1.4
Proportionate	Supports selective acquisition methods that can be tailored to the aims of the investigation – see sections 5.1.2 to 5.1.4.
Training / Education aid	The comprehensive auditing could be used as a training aid and to enhance best practice. See sections 4.3 and 5.2.
Efficiency – processing / searching / backup.	DEB EUs permit the structured decomposition and grouping of similar types of information – see sections 3.3.1.2 and 3.5.1.7. The DEB framework supports selectable compression, and encryption algorithms – see sections 3.5.1.9 and 3.5.1.10.
Tool certification / testing	The DEB API provides a mechanism that could be used for assurance and certification of digital forensic applications – see section 3.6.
Multi-investigator support	The TCB could be used for this – see section 3.3.1.4.
Support for disparate range of digital devices and environments.	This is demonstrated in the various scenarios discussed in Chapter 5

Table 4 - DEB Framework & Requirements Correlation

7.2 Recommendations for Further Work

When compared with other storage methods DEBs provide high flexibility and a number of features that are not implemented by other methods. A number of additional features could be added that would further enhance the framework. The following features are suggested for incorporation into the DEB framework definition:

- Integrate a public key cryptographic system within the DEB framework. This would enhance confidentiality and restrict access to authorised personnel. The current definition only supports symmetric key encryption methods for the index and bag file contents.

- Implement an error correction system to automatically repair the contents of a DEB should a component become corrupt. Error detection is already possible through the use of cryptographic digests embedded within the tag and index components. This could be further extended using error correction technologies similar to that incorporated into the parity archive system that was developed for Usenet data transfer - http://parchive.sourceforge.net/. To accommodate this method an Evidence Unit could be used that contains all the information required to rebuild any other EU within the DEB. However, further work is required to validate this approach. Any self-repairing process that was carried out should also be logged.

- Further consideration is required for the mechanism by which DEB components can be accessed or distributed in a parallel or grid processing environment.

- Similarly, further development is required to extend the DEB analysis protocol in order to maintain the integrity and reliability of DEB transactional processing. This specifies the sequence that a DEB analysis application should follow to access the contents of a DEB. In addition, the ability to add new EUs by the analysis application that contain distilled or analysed information as opposed to simply extracting data from a DEB into 'plain' files or another DEB.

- Although the concept of 'Intelligent Imaging' is supported by the DEB framework, further consideration is required into how investigative domain expertise could be captured and incorporated into an intelligent imaging system.

- Another feature that requires further consideration is the mechanism by which media acquisition errors are recorded within a DEB.

- The process by which a DEB bag file can contain another DEB or pointers to other bags is also required.

- Further consideration and guidance is required to formalise the protocol that is used in order for new and disparate devices to utilise the DEB framework. This thesis demonstrates how DEBs incorporate traditional digital forensic scenarios such as device imaging, command line utility capture and even credit card skimming devices. A formal methodology and process for the registration and documentation for incorporating disparate devices in a standardised way into DEBs aids interoperability between independently developed acquisition and analysis applications, and thus permits formal certification and verification of those applications. The requirement for an official registration facility for the registration of DEB content, integrity, compression, encryption and Meta-Tag types was identified in section 3.5 of this thesis. However, if this is extended to incorporate formal registration, certification and testing of digital forensic products then this would significantly advance the scientific rigor and maturing of digital forensics as a forensic science.

In summary, the DEB framework provides the majority of features expected of any digital evidence storage format and provides a more flexible and extensible system than any other format. Some degree of future-proofing of the DEB framework may be gained by the implementation of some of the features suggested in the further work discussion.

The challenge now remains to get international adoption of the DEB framework and wide-scale deployment.

8 References

ACPO (2003) *Good Practice Guide for Computer Based Electronic Evidence*. Version 3, NH3378, Association of Chief Police Officers (ACPO), National Hi-Tech Crime Unit (NHTCU), August 2003.

ACPO (2005) *Practice Advice on Investigating Indecent Images of Children on The Internet*, 2005, ACPO, National Centre for Policing Excellence. Available from: http://www.kent.police.uk/About%20Kent%20Police/Policy/pdfs/n85%20acpo.pdf [Accessed 19 March 2007]

ANSI (2005) *AT Attachment – 7 with Packet Interface (ATA/ATAPI-7),* ANSI NCITS 397-2005.

ASR (2002) *Expert Witness Compression Format Specification*, April 7, 2002, ASR Data. Available from: http://www.asrdata.com/SMART/whitepaper.html [Accessed 19 March 2007]

Bunting (2006) *EnCase Certified Examiner Study Guide,* Steve Bunting, William Wei, Wiley Publishing Inc., 2006. ISBN 0-7821-4435-7. Chapter 5 EnCase Concepts – EnCase Evidence File Format.

Butler (2005) *Forensic DNA Typing : Biology, Technology, and Genetics of STR Markers (2nd Edition)* – J. Butler 2005. Elsevier Academic Press.

Carrier (2005) *File System Forensic Analysis*, Brian Carrier, Addison-Wesley Publishing, 2005, ISBN 0-321-26817-2.

Carvey(2001) *NT/2K Incident Response Tools* – H. Carvey 2001-08-15 http://www.securityfocus.com/infocus/1294 [Accessed 9 September 2007]

Casey (2004) Digital Evidence and Computer Crime – Second Edition. Eoghan Casey. Elsevier Academic Press. 2004. ISBN : 0-12-163104-4.

CDESF (2006) *Standardizing Digital Evidence Storage*. The Common Digital Evidence Storage Format Working Group. Communications of the ACM – February 2006, Volume 49. No. 2, Pages 67- 68.

Coulouris et al.(2000) *Distributed Systems – Concepts and Design – 3rd Edition*, George Coulouris, Jean Dollimore and Tim Kindberg, Addison Wesley. 2000, ISBN 0-201-61918-0.

DFRWS (2001) *A Roadmap for Digital Forensic Research*, Digital Forensic Research WorkShop Technical Report, DTR-T001-01, November 2001. Available from: http://www.dfrws.org/2001/dfrws-rm-final.pdf [Accessed 19 March 2007]

DFRWS (2006) *Survey of Disk Image Storage Formats*, Version 1, September 1 2006, DFRWS Common Digital Evidence Storage Format working group. Available from: http://www.dfrws.org/CDESF/survey-dfrws-cdesf-diskimg-01.pdf [Accessed 19 March 2007]

DOD (2007) *DOD Computer Forensic Laboratory (DCFL) dd utility*. Available from: http://dcfldd.sourceforge.net/ [Accessed 18 March 2007].

DOJ (2003) *Analysing DNA evidence*. Department of Justice Available from: http://www.dna.gov/basics/analysis/ [Accessed 19 November 2007]

Duce, Mitchell & Turner (2007) *Digital Forensics: Challenges and Opportunities*, D.A. Duce, F. R. Mitchell & P.Turner, Proceedings of 2nd Conference on Advances in Computer Security and Forensics (ACSF), Liverpool John Moores University – 12-13 July 2007, Liverpool UK.

EWCA (2002) R v Oliver, Hartrey, Baldwin (2003) – Annex B [2002] EWCA Crim 2766 Available from: http://www.gmc-uk.org/about/council/papers/2004_03/6c_annex_b.pdf [Accessed 18 March 2007].

Farmer & Venema (2005) *Forensic Discovery*, Dan Farmer & Wietse Venema, Addison Wesley, 2005, ISBN 0-201-63497-X.

Forte (2004) *The "ART" of log correlation - Tools and Techniques for Correlating Events and Log files* - Dario Valentino Forte, Computer Fraud & Security, Issue 8, Pages 15-17, August 2004.

Foundstone(2007) Foundstone whitepapers and tools - http://www.foundstone.com/us/resources-whitepapers.asp http://www.foundstone.com/us/resources-free-tools.asp [Accessed 9 September 2007].

German (2007) *The History of Fingerprints* – E. German, 2007. Available from: http://www.onin.com/fp/fphistory.html [Accessed 19 November 2007]

Helix(2007) Helix Live CD page http://www.e-fense.com/helix/ [Accessed 9 September 2007].

IETF (1992) *The MD5 Message-Digest Algorithm*, RFC 1321, Internet Engineering Task Force. Available from: http://www.faqs.org/rfcs/rfc1321.html [Accessed 18 March 2007].

IETF (1997) *Key words for use in RFCs to Indicate Requirement Levels*, RFC 2119, Internet Engineering Task Force. Available from: http://www.faqs.org/rfcs/rfc2119.html [Accessed 18 March 2007].

IETF (2001) *US Secure Hash Algorithm 1 (SHA1),* RFC 3174, Internet Engineering Task Force. Available from: http://www.faqs.org/rfcs/rfc3174.html [Accessed 18 March 2007].

IETF (2002) *Date and Time on the Internet: Timestamps*, Internet Engineering Task Force – RFC 3339, July 2002. Available from http://www.ietf.org/rfc/rfc3339.txt [Accessed 3 July 2007]

ISO/IEC 4909(2006) *Identification Cards - Financial Transaction Cards - Magnetic stripe data content for track 3*, International Organisation for Standardisation, 2006.

ISO/IEC 7810(2003) *Identification Cards – Physical Characteristics*, International Organisation for Standardisation, 2003.

ISO/IEC 7811-X(2001-2004) *Identification Cards – Recording techniques*, International Organisation for Standardisation, 2001 – 2004.

ISO/IEC 7813(2006) *Identification Cards - Financial Transaction Cards*, International Organisation for Standardisation, 2006.

Kenneally(2005a) *Risk sensitive digital evidence collection*. Erin Kenneally, Chris Brown -Digital Investigation - Elsevier. Volume 2 Number 2, June 2005. Elsevier.

Kenneally(2005b) *Realizing Risk Sensitive Evidence Collection*. Erin Kenneally, Chris Brown. Digital Forensic Research Workshop (DFRWS). 2005 proceedings. http://www.dfrws.org/2005/proceedings/keneally_risk_slides.pdf [Accessed 9 September 2007].

Kovacich & Jones (2006) *High-Technology Crime Investigator's Handbook – Second Edition*, Dr. Gerald L. Kovacich, Dr. Andy Jones, Elsevier. 2006, ISBN 13: 978-0-7506-7929-9.

Masters & Turner (2007) *Forensic data recovery and examination of magnetic swipe card cloning devices*, G.Masters & P.Turner, Digital Investigation – Elsevier. Volume 4S, Pages S16-S22, doi:10.1016/j.diin.2007.06.018

Microsoft (2000) *Logical Block Addresssing (LBA) Defined*, Article ID 122052. Microsoft. Available from: http://support.microsoft.com/kb/122052/EN-US/ [Accessed 18 March 2007].

Microsoft (2003) *How To : Windows 2000 Assigns, Reserves and Stores Drive Letters*, Microsoft Article ID : 234048 - Revision 1.0, November 21, 2003.

Microsoft (2004) *Order in Which MS-DOS and Windows Assign Drive Letters*, Microsoft Article ID : 51978 – Revision 2.1, September 28, 2004.

Microsoft (2007a) *Windows Support for Large IDE Hard Disks*, Article ID 126855, Microsoft. Available from: http://support.microsoft.com/kb/126855/en-us [Accessed 18 March 2007].

Microsoft (2007b) *Description of NTFS date and time stamps for files and folder*, Article ID 299648, Microsoft. Available from: http://support.microsoft.com/kb/299648 [Accessed 18 March 2007].

Microsoft (2007c) How *To: Manage Drives and partitions Windows 2000*, Microsoft Article ID : 323967 – , Revision 3.4, May 7, 2007.

Microsoft (2007d) *Microsoft Windows XP Command line reference A-Z* http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/ntcmds.mspx?mfr=true [Accessed 9 September 2007].

Microsoft (2007e) *OFF97: Invalid Page Fault in Ole32.dll Closing File*, Microsoft Article ID 180188, Rev 3.1, January 22, 2007. http://support.microsoft.com/kb/q180188/ [Accessed 15 September 2007].

Microsoft (2007f) *Microsoft MSDN Library IIS 6.0 SDK – IIS Log File Formats* http://msdn2.microsoft.com/en-us/library/ms525807.aspx [Accessed 10 September 2007].

Microsoft (2007g) *Microsoft SMTP Transport Architecture - protocol logging, event logging and message tracking* http://www.Microsoft.com/technet/prodtechnol/exchange/guides/E2k3TechRef/d620b29f-a307-4f4f-a169-a1cd02afb642.mspx [Accessed 10 September 2007].

Mitchell (1998) *An introduction to Knowledge Acquisition*, F. Mitchell, July 1998. Oxford Brookes University, School of Computing & Mathematical Sciences, CMS-TR-98-06.

OSTA (2003) *Understanding CD-R & CD-RW,* Optical Storage Technology – Revision 1.00, 2003. Available from http://www.osta.org/technology/cdqa13.htm [Accessed 3 July 2007].

Pyrek (2007) *Forensic Science Under Siege*, Kelly M. Pyrek, Elsevier Academic Press, 2007, ISBN-13 : 978-0-12-370861-8.

Rowlinson (2004) *A Ten Step Process for Forensic Readiness*, Robert Rowlinson, International Journal of Digital Evidence, Winter 2004, Volume 2, Issue 3. Available from: http://www.utica.edu/academic/institutes/ecii/publications/articles/A0B13342-B4E0-1F6A-156F501C49CF5F51.pdf , www.ijde.org [Accessed 3 July 2007]

Sammes & Jenkinson (2000) *Forensic Computing : A Practioner's Guide*, Tony Sammes and Brian Jenkinson, Springer-Verlag, 2000, ISBN 1-85233-299-9.

Schatz (2006) *A correlation method for establishing provenance of timestamps in digital evidence*, Bradley Schatz, George Mohay, and Andrew Clark Digital Investigation, Elsevier. Volume 3S. Pages S98-S107. 2006.

Schiffman (2001) *The Hackers Challenge : Test Your Incident Response Skills Using 20 Scenarios*, Mike Schiffman. Osborne/Mc Graw-Hill, ISBN 0-07-219384-0, 2001.

Sorenson (2003a) *Incident Response Tools For Unix, Part One: System Tools* – Holt Sorenson 2003-03-27 www.securityfocus.com/infocus/1679 [Accessed 9 September 2007].

Sorenson (2003b) *Incident Response Tools For Unix, Part Two: File-System Tools* – Holt Sorenson 2003-10-17 www.securityfocus.com/infocus/1738 [Accessed 9 September 2007].

Steel (2006) *Windows Forensics: The Field Guide for Conducting Corporate Computer Investigations*, Chad Steel, Wiley Publishing, 2006, ISBN 0-470-03862-4.

Stephenson (2003) *A Comprehensive Approach To Digital Incident Investigation*, Peter Stephenson, Elsevier Information Security Technical Report, 2003. Available from: http://www.emich.edu/cerns/downloads/pstephen/Comprehensive-Approach-to-Digital-Investigation.pdf [Accessed 26 February 2008].

Stephenson (2004) *Structured Investigation of Digital Incidents in Complex Computing Environments*, Peter Stephenson, PhD Thesis, Oxford Brookes University, October 2004.

Thinkquest (2004) *Forensic Science – Fingerprinting History* Available from: http://library.thinkquest.org/04oct/00206/nts_fingerprinting.htm [Accessed 19 November 2007].

Thompson & Best (2000) *The future of magnetic data storage technology* – D.A. Thompson & J. S. Best. IBM Journal of Research and Development, Volume 44, Number 3.
Available from: http://www.research.ibm.com/journal/rd/443/thompson.html [Accessed 19 November 2007].

TSO (1861) *Offences Against the Person Act 1861)*, The Stationery Office Ltd (formerly Her Majesty's Stationery Office (HMSO)).

TSO (1957) *Homicide Act 1957*. The Stationery Office Ltd (formerly Her Majesty's Stationery Office (HMSO)).

TSO (1967) *Criminal Law Act 1967*, The Stationery Office Ltd (formerly Her Majesty's Stationery Office (HMSO)).

TSO (1968) *Theft Act 1968, s25*, The Stationery Office Ltd (formerly Her Majesty's Stationery Office (HMSO)).

TSO (1990) *Computer Misuse Act 1990 (c.18)*, The Stationery Office Ltd (formerly Her Majesty's Stationery Office (HMSO)). Available from: http://www.opsi.gov.uk/acts/acts1990/Ukpga_19900018_en_1.htm [Accessed 19 March 2007]

TSO (2000) *Regulation of Investigatory Powers Act 2000 (c.23)*, The Stationery Office Ltd (formerly Her Majesty's Stationery Office (HMSO)). Available from: http://www.opsi.gov.uk/acts/acts2000/pdf/ukpga_20000023_en.pdf [Accessed 9 February 2008]

TSO (2006) *Police and Justice Act 2006 (c.48)*, The Stationery Office Ltd (formerly Her Majesty's Stationery Office (HMSO)). Available from: http://www.opsi.gov.uk/acts/acts2006/pdf/ukpga_20060048_en_1.htm [Accessed 9 February 2008]

Turner (2005a) *Digital provenance – interpretation, verification and corroboration*, Philip Turner, Digital Investigation – Elsevier, Volume 2, Pages 45-49 doi:10.1016/j.diin.2005.01.002

Turner (2005b) *Unification of digital evidence from disparate sources (Digital Evidence Bags)*, Philip Turner, Digital Investigation – Elsevier, Volume 2, Pages 223-228, doi:10.1016/j.diin.2005.07.001

Turner (2006) *Selective and Intelligent imaging using digital evidence bags*, Philip Turner, Digital Investigation – Elsevier, Volume 3S, Pages S59-S64, doi:10.1016/j.diin.2006.06.003

Turner (2007a) *Digital Evidence Provenance and Continuity using Digital Evidence Bags*, Philip Turner, Proceedings of 2nd Conference on Advances in Computer Security and Forensics (ACSF), Liverpool John Moores University – 12-13 July 2007, Liverpool UK.

Turner (2007b) *Applying a forensic approach to incident response, network investigation and system administration using Digital Evidence Bags*, Philip Turner.

Digital Investigation – Elsevier, Volume 4, Pages 30-35, doi:10.1016/j.diin.2007.01.002

Wang & Yu (2005) *How to Break MD5 and Other Hash Functions*, Xiaoyun Wang and Hongbo Yu, Advances in Cryptology – Eurocrypt 2005, 24th Annual Conference on the Theory and Applications of Cryptographic Techniques, Springer 2005. ISBN 978-3-540-25910-7, http://dx.doi.org/10.1007/11426639_2

Wetstone (2007) *Digital Evidence Time Stamping (DETS)*, Wetstone. Available from http://www.wetstonetech.com/f/DS_DETS.pdf [Accessed 3 July 2007] http://www.wetstonetech.com/f/faq_dets.html [Accessed 3 July 2007]

Library of links to useful tools and utilities:

http://www.antihackertoolkit.com/tools.html -compiled from the book Anti Hacker Toolkit by Keith Jones, Mike Shema, Bradley Johnson. McGraw-Hill Osbourne Media, June 2002. ISBN 0072222824. [Accessed 9 September 2007]

Free Microsoft tools – Windows 2000 Resource Kit for Administative Tasks http://support.Microsoft.com/kb/927229 [Accessed 9 September 2007]

8.1 Conferences Attended and Presentations Given

During this course of study the following conferences have been attended the author has either presented a paper or been a panel member to lead discussion.

2003 – Digital Forensic Tools and Technology – Panel Member
Digital Forensic Research Workshop (DFRWS) conference 2003, 6-8th August, Cleveland, Ohio, USA.

2004 – Challenges to the adaptation of current digital forensic practice to the operational realm - Panel Member
Digital Forensic Research Workshop (DFRWS) conference 2004, 11-13th August, Baltimore, Maryland, USA.
Available from :
www.dfrws.org/2004/day2/Day2-Panel-Discussion-v2.ppt [Accessed 9 September 2007]

2005 – Research Needs of Law Enforcement – Panel Member

Digital Forensic Research Workshop (DFRWS) conference 2005, 17-19th August, New Orleans, Louisiana, USA.

2005 - Unification of digital evidence from disparate sources (Digital Evidence Bags)

Digital Forensic Research Workshop (DFRWS) conference 2005, 17-19th August, New Orleans, Louisiana, USA.

Available from :

www.dfrws.org/2005/proceedings/turner_evidencebags.pdf

www.dfrws.org/2005/proceedings/turner_evidencebags_slides.pdf

[Accessed 9 September 2007]

2006 - Selective and Intelligent imaging using digital evidence bags

Digital Forensic Research Workshop (DFRWS) conference 2006, 14-16th August, Lafayette, Indiana, USA.

Available from :

www.dfrws.org/2006/proceedings/8-Turner.pdf

www.dfrws.org/2006/proceedings/8-Turner-pres.pdf

[Accessed 6 September 2007]

2006 – Digital Evidence Bags

F3 Annual Conference 25-26 October 2007, Gloucestershire, UK.

2007 Digital Evidence Provenance and Continuity using Digital Evidence Bags

2nd Conference on Advances in Computer Security and Forensics (ACSF)

Liverpool John Moores University – 12-13 July 2007. Liverpool. UK

2007 Forensic data recovery and examination of magnetic swipe card cloning devices *Digital Forensic Research Workshop (DFRWS) conference 2007, 13th-15th August, Pittsburgh, PA, USA.*
Available from :
www.dfrws.org/2007/proceedings/p16-masters.pdf
[Accessed 6 September 2007]

9 Appendix A – Published papers

9.1 Digital provenance – interpretation, verification and corroboration

9.2 Unification of digital evidence from disparate sources

9.3 Digital Evidence Provenance and Continuity using DEBs

9.4 Selective and Intelligent imaging using digital evidence bags

9.5 Applying a forensic approach to incident response, network investigation and system administration using Digital Evidence Bags

9.6 Standardizing Digital Evidence Storage - The Common Digital Evidence Storage Format Working Group

9.7 Forensic data recovery and examination of magnetic swipe card cloning devices

9.8 Digital Forensics : Challenges and Opportunities

MISSING PAGES REMOVED ON INSTRUCTION FROM THE UNIVERSITY

11 Appendix C - Patent application

The DEB framework and concept has also formed the basis of a patent application sponsored by QinetiQ (Appendix D). The date of filing of this application was 25 May 2005, application number 0510878.2 and been moved forward for international filing with International application number PCT/GB2006/001942.

The current status of the application can be found at the World Intellectual Property Organisation web site:
www.wipo.int/pctdb/en/wo.jsp?wo=2006126006&IA=WO2006126006&DISPLAY=STATUS [Accessed 9 September 2007]

The claims of the patent application are:

1. A method of capturing digital data, the method comprising the steps of:

- *copying digital data from a data source into one or more evidence files;*
- *for each evidence file recording data descriptive of at least one of the source and the contents of the digital data in the evidence file;*
- *recording, in a tag file, data indicative of provenance of the digital data in the one or more evidence files.*

2. A method according to any preceding claim in which digital data is copied into a plurality of evidence files.

3. A method according to any preceding claim in which the digital data is selectively copied from the data source into the one or more evidence files.

4. A method according to any preceding claim in which, for each evidence file, the data descriptive of one of the source and contents of the digital data is stored in an index file distinct from the evidence file.

5. A method according to claim 4 in which a distinct index file is created for each evidence file.

6. A method according to any preceding claim in which at least the data descriptive of one of the source and contents of the digital data comprises a digital fingerprint of the digital data.

7. A method according to any preceding claim in which the tag file comprises a digital fingerprint of at least one of the evidence files.

8. A method according to any preceding claim in which the tag file comprises a description of the format of the data descriptive of one of the source and contents of the digital data.

9. A method according to any preceding claim in which the data source is a data storage medium.

10. A method according to any preceding claim in which the data source is a data transmission medium.

11. A method according to any preceding claim in which multiple indications of provenance are associated with at least one given item of the digital data in the one or more evidence files.

12. A program for a computer having respective code portions and data structures to perform the steps of the methods of any one of claims 1 - 11.

13. Apparatus for capturing digital data, the apparatus comprising:

- *means for copying digital data from a data source into one or more evidence files;*
- *for each evidence file, means for recording data descriptive of at least one of the source and the contents of the digital data in the evidence file;*
- *means arranged to record, in a tag file, data indicative of provenance of the digital data in the one or more evidence files.*

14. A data structure for capturing digital data, the data structure comprising:

- *at least one evidence file for containing digital data copied from a data source;*
- *at least one index file containing data descriptive of at least one of the source and contents of the digital data in the at least one evidence files;*
- *a tag file containing data indicative of provenance of the digital data in the at least one evidence files.*

15. A method of accessing a data structure according to claim 13, the method comprising the steps of:

- *identifying one or more evidence files to be accessed;*
- *recording details of the evidence file access in the tag file of the data structure;*
- *recording a new integrity check value in the tag file, responsive to the contents of the tag file including the newly-recorded details of the evidence file access.*

16. A method according to claim 15 in which the details of the evidence file access comprise at least one of:

- *identification of the application performing the evidence file access;*
- *identification of the user requesting evidence file access;*
- *identification of the time of evidence file access;*

17. A method according to any one of claims 16-17 in which the integrity check is a digital fingerprint .

18. A method according to claim 17 in which the digital fingerprint is one of a CRC digits, an MD5 hash, and a SHA hash.

19. Apparatus for accessing a data structure according to claim 14, the apparatus comprising:

- *means for identifying one or more evidence files to be accessed;*

- *means for recording details of the evidence file access in the tag file of the data structure;*
- *means for recording a new integrity check value in the tag file, responsive to the contents of the tag file including the newly-recorded details of the evidence file access.*

20. A method of updating a data structure according to claim 14, the method comprising the steps of:

- *accessing the data structure to extract evidential data contained within it;*
- *processing evidential data extracted from the data structure to create a new evidence file and corresponding index file;*
- *adding the new evidence file and index file to the existing data structure;*
- *appending continuity information to the tag file of the data structure indicative of the addition of the new evidence file and index file.*

21. Apparatus for updating a data structure according to claim 14, the apparatus comprising:

- *means for accessing the data structure to extract evidential data contained within it;*
- *means for processing evidential data extracted from the data structure to create a new evidence file and corresponding index file;*
- *means for adding the new evidence file and index file to the existing data structure;*
- *means for appending continuity information to the tag file of the data structure indicative of the addition of the new evidence file and index file.*

22. Data structures, methods, apparatus, systems, and programs for computers for digital data capture and processing substantially as described in the foregoing specification and with reference to the accompanying drawings.