

# **VALIDATING DIGITAL FORENSIC EVIDENCE**

**A THESIS PRESENTED**

**BY**

**Karthikeyan Shanmugam**

SEPTEMBER 2011

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of  
Philosophy

Electronic and Computer Engineering  
School of Engineering, Design and Technology  
Brunel University Uxbridge UB8 3PH, UK

## **Dedication**

This work is dedicated to my parents T. Shanmugam and Mallika Shanmugam whom together made the required sacrifice to ensure that my siblings and I got university education and career.

**BRUNEL UNIVERSITY**

**VALIDATING FORENSIC EVIDENCE AN ANTI-FORENSIC APPROACH**

**Declaration:** *I have read and I understand the Department's guidelines' on plagiarism and cheating, and I certify that this submission fully complies with these guidelines.*

**Student's name:** **KARTHIKEYAN SHANMUGAM**

**Signature of student:**

## **ABSTRACT**

This dissertation focuses on the forensic validation of computer evidence. It is a burgeoning field, by necessity, and there have been significant advances in the detection and gathering of evidence related to electronic crimes. What makes the computer forensics field similar to other forensic fields is that considerable emphasis is placed on the validity of the digital evidence. It is not just the methods used to collect the evidence that is a concern. What is also a problem is that perpetrators of digital crimes may be engaged in what is called anti-forensics. Digital forensic evidence techniques are deliberately thwarted and corrupted by those under investigation. In traditional forensics the link between evidence and perpetrator's actions is often straightforward: a fingerprint on an object indicates that someone has touched the object. Anti-forensic activity would be the equivalent of having the ability to change the nature of the fingerprint before, or during the investigation, thus making the forensic evidence collected invalid or less reliable. This thesis reviews the existing security models and digital forensics, paying particular attention to anti-forensic activity that affects the validity of data collected in the form of digital evidence. This thesis will build on the current models in this field and suggest a tentative first step model to manage and detect possibility of anti-forensic activity. The model is concerned with stopping anti-forensic activity, and thus is not a forensic model in the normal sense, it is what will be called a “meta-forensic” model. A meta-forensic approach is an approach intended to stop attempts to invalidate digital forensic evidence. This thesis proposes a formal procedure and guides forensic examiners to look at evidence in a meta-forensic way.

## **ACKNOWLEDGEMENTS**

I would like to thank number of people who gave me the inspiration and strength to complete this work.

I thank Dr Roger Powell and Dr Thomas Owens for their commitment, enthusiasm and motivation which inspired me to complete this thesis. Dr Powell laid a strong foundation shaping me to think like an academic while Dr Owens nurtured my ideas which helped me complete this thesis. Their invaluable guidance, comments and constructive criticisms have been crucial in helping me to complete the PhD work there are no words to thank them for their support.

I would also like to thank all my friends who have directly or indirectly prayed, helped, supported and encouraged me throughout this work.

Thanks to my lovely son Varun Karthik whose love has continued to be a source of inspiration to me in all that I do. Very special thanks to my wife Akilandeswari Chandrasekar for providing me with moral support whenever I had difficult time with my research. Her contribution in looking after the family affairs gave me time to finish this project.

Finally, I would like to thank the Almighty God for making it possible for me to accomplish one of my childhood ambitions.

## Table of Contents

<b>ABSTRACT.....</b>	<b>III</b>
<b>ACKNOWLEDGEMENTS.....</b>	<b>IV</b>
<b>LIST OF TABLES.....</b>	<b>V</b>
<b>LIST OF FIGURES.....</b>	<b>VI</b>
<b>LIST OF ACRONYMS AND ABBREVIATIONS.....</b>	<b>XIII</b>
<b>CHAPTER 1: INTRODUCTION.....</b>	<b>1</b>
1.1 THE HYPOTHESIS.....	3
1.2 THE BASICS OF FORENSIC INVESTIGATION.....	5
1.3 ANTI-FORENSICS.....	5
1.4 RESEARCH AIMS AND OBJECTIVES.....	6
1.5 DISSERTATION STRUCTURE.....	7
1.6 SUMMARY OF RESEARCH ACHIEVEMENTS.....	9
<b>CHAPTER 2: CONCEPTS OF DIGITAL FORENSICS AND ANTI-FORENSICS.....</b>	<b>11</b>
2.1 THE PROCESS.....	13
2.1.1 EVIDENCE.....	14
2.1.2 CLASSES OF DIGITAL EVIDENCE.....	16
2.1.3 ANONYMITY OF DIGITAL INFORMATION.....	17
2.1.4 STANDARD AND BURDEN OF PROOF .....	17
2.1.5 CONTEXT OF DIGITAL INFORMATION.....	18
2.1.6 LEGAL FRAMEWORK.....	18
2.2 EVALUATING THE CASE.....	20
2.3 RECONSTRUCTION AND VALIDATION.....	21
2.3.1 DEVICE SETUP ANALYSIS.....	23

2.3.2	FILE SYSTEM ANALYSIS.....	23
2.3.3	SYSTEM LOG FILE ANALYSIS.....	26
2.3.4	VOLATILE MEMORY ANALYSIS.....	27
2.4	ANTI-FORENSIC CONCEPTS.....	29
2.5	ANTI-FORENSIC METHODS .....	29
2.5.1	DATA HIDING.....	30
2.5.2	ARTIFACT WIPING.....	31
2.5.3	TRAIL OBFUSCATION.....	32
2.5.4	ATTACKS AGAINST TOOLS.....	32
2.6	METASPOILT PROJECT.....	33
2.7	ANTI-FORENSIC INTIAL COMPROMISE.....	35
2.8	FUZZY LOGIC.....	35
2.9	SUMMARY.....	38
	<b>CHAPTER 3: LITERATURE REVIEW OF MODELS AND PROCESS.....</b>	<b>40</b>
3.1	MODELLING SECURITY AND FORENSIC BEHAVIOUR.....	41
3.2	APPROACH TO EVIDENCE IN CYBERSPACE.....	41
3.3	THREE-LEVEL HIERARCHICAL MODEL.....	43
3.4	FIRST RESEARCH ROAD MAP (DFRWS).....	45
3.4.1	NEW DIGITAL FORENSIC SCIENCE DEFINITION.....	45
3.5	AN ABSTRACT PROCESS MODEL.....	48
3.6	END-TO-END DIGITAL INVESTIGATION PROCESS.....	49
3.7	EVENT-BASED DIGITAL FORENSIC FRAMEWORK.....	50
3.8	CASE-RELEVANCE INFORMATION INVESTIGATION FRAMEWORK.....	51
3.9	FORENSIC MODELLING COMPARISON AND OUTCOME.....	52

3.10 POLICY BASED SECURITY MODEL.....	54
3.11 BELL-LAPADULA MODEL.....	55
3.12 BIBA MODEL.....	55
3.13 TAKE-GRANT PROTECTION MODEL.....	56
3.14 SECURITY MODEL COMPARISON AND OUTCOME.....	56
3.15 MODELLING NETWORK FORENSIC PROCESSES.....	57
3.16 VULNERABILITY TREE MODEL.....	59
3.17 REASONING AND LOGIC.....	60
3.17.1 EXTENDED FINITE STATE MACHINE.....	60
3.17.2 COUNTERFACTUAL REASONING.....	62
3.17.3 ATTACK TREES AND VULNERABILITY TREE REASONING.....	64
3.17.4 WHY-BECAUSE ANALYSIS .....	65
3.18 MULTI LINEAR EVENTS SEQUENCING (MES).....	66
3.19 SUMMARY.....	67
<b>CHAPTER 4: THE ANTI -FORENSIC EXPERIMENT .....</b>	<b>69</b>
4.1 COMPONENTS OF THE EXPERIMENT DEMONSTRATING ANTI-FORENSIC DATA COMPROMISE.....	69
4.2 TESTING FRAMEWORK.....	70
4.2.1 NIST CFTT.....	71
4.2.2 BRIAN CARRIER’S DIGITAL FORENSIC TOOL TESTING (DFTT).....	71
4.2.3 COMBINATION METHOD.....	72
4.3 COMBINATION METHOD IMPLEMENTATION.....	72



4.4 THE EXPERIMENTAL SETUP.....	73
4.4.1 SECURE REMOVE (SRM) AND TESTDISK UTILITY.....	74
4.4.2 ENCASE EVIDENCE VERIFICATION.....	77
4.4.3 THE CORONERS TOOLKIT.....	81
4.4.4 THE NETWORK AF.....	82
4.5 EVIDENCE IN RELATION TO AF.....	89
4.6 SUMMARY.....	90
<b>CHAPTER 5: THE META-FORENSIC MODEL.....</b>	<b>91</b>
5.1 CLASSIFICATION.....	91
5.2 DECISION TREE.....	93
5.2.1 LEVEL DEPTH.....	95
5.3 THE MEASUREMENT AND PERCEPTION BASED INFORMATION.....	96
5.4 ASSIGNING WEIGHTS.....	98
5.4.1 NODE VALUE AFTER ASSIGNING WEIGHTS.....	106
5.4.2 FUZZY UNIVERSAL SET.....	107
5.5 OPERATIONS.....	108
5.6 MODELLING THE ANTI-FORENSIC ACTIVITY.....	110
5.7 MODEL APPLICATION CASE1 .....	116
5.8 MODEL APPLICATION CASE2.....	125
5.8.1 THE ISSUE.....	126
5.8.2 META-FORENSIC APPROACH.....	127
5.9 SYSTEM AUTOMATION.....	134
5.10 SUMMARY.....	137

<b>CHAPTER 6: CONCLUSION AND FUTURE WORK.....</b>	<b>138</b>
REFERENCES.....	142
APPENDIX A.....	149
APPENDIX B.....	151
APPENDIX C.....	153
APPENDIX D.....	182
APPENDIX E.....	185

## LIST OF TABLES

Table 1: Forensic models & the investigation steps.....	53
Table 2: Files for Experiment.....	78
Table 3: System Scan Reports.....	85
Table 4: Anti-Forensic Categories.....	91
Table 5: Logically assigning weights.....	103
Table 6: Logically assigning weights for data hiding.....	122
Table 7: Logically assigning weights for artefact wiping.....	125

## LIST OF FIGURES

Figure 1: Diagrammatic representation of investigation of hypothesis.....	4
Figure 2: Reconstruction and validation of primary object of analysis.....	23
Figure 3: List of process and memory usage in Windows.....	28
Figure 4: List of process and memory usage in Unix.....	28
Figure 5: Evidence Process.....	42
Figure 6: Forensic Evidence Path.....	42
Figure 7: A Three-Level Hierarchical Model for Forensic Evidence.....	43
Figure 8: Investigative Process for Digital Forensic Science.....	47
Figure 9: Categories of Data Hiding.....	48
Figure10: General Computer Network Forensics System Architecture.....	58
Figure 11: Finite State Transition.....	61
Figure 12: EFSM representing Boolean conditions.....	62
Figure 13: MES Diagram for validating data.....	63
Figure 14: Attack tree describing different ways to open a safe.....	64
Figure 15: Additional elements for MES.....	67
Figure 16: Experiment Setup for anti-forensic activity.....	74
Figure 17: Secure Remove process.....	76
Figure 18: Evidence Verification.....	79
Figure 19: Vulnerability Exploit.....	87
Figure 20: Experimental Result as MES Diagram.....	88
Figure 21: Anti-forensic activity tree.....	95

Figure 22: Node information classification.....	97
Figure 23: Assigning Weights for nodes.....	99
Figure 24: WB reasoning for a node.....	101
Figure 25: Working model showing Anti-forensic activity for a level 3 node.....	112
Figure 26: Anti-forensic activity result for level 3 nodes... ..	115
Figure 27: Anti-forensic activity result for all nodes.....	116
Figure 28: Reasoning results.....	123
Figure 29: Anti-forensic activity result for level 1, level 2 and level 3.....	124
Figure 30: Architecture for application of Meta-forensic Model.....	126
Figure 31: Obfuscation method's sub category.....	128
Figure 32: System Automation of Meta-Forensic Validity Model.....	136

## LIST OF ACRONYMS AND ABBREVIATIONS

<b>Terms</b>	<b>Meanings</b>
ACPO	Association of Chief Police Officers
AF	Anti-Forensics
AV	Anti Virus
CIM	Common Information Model
CFTT	Computer Forensic Tool Testing
DFRWS	Digital Forensic Research Workshop
DOD	Department of Defence
DFTT	Digital Forensic Tool Testing
DMTF	Distributed Management Task Force
EB	Event Building Bloc
EDDI	End-to-End Digital Investigation Process
EFSM	Extended finite state machine
FAT	File Allocation Table
FTP	File Transfer protocol
FAM	Finite State Machine
HFS	Hierarchical File System
HTTP	Hyper Text Transfer Protocol
IDIP	Integrated Digital Investigation Process
IP	Internet Protocol

IDS	Intrusion Detection System
MB	Measurement Based
MBID	Model Based Intrusion Detection
MES	Multi Linear Events Sequencing
NBS	National Bureau of Standards
NFS	Network File System
NIST	National Institute of Standards and Technology
NIDS	Network Intrusion Detection System
NSA	National Security Agency
NTFS	New Technology File System
OBSP	Object Based Sub Phases
PB	Perception Based
PDA	Personal Digital Assistant
PGP	Pretty Good Privacy
RAM	Random Access Memory
SEE	Survey Extract Examine
SMB	Server Message Block
TAME	Threat Assessment Methodology
TCP/IP	Transmission Control Protocol/Internet Protocol
TCG	Trusted Computing Group
TIF	Trusted Internet Forensics

TLA	Temporal Logic of Action
UDP	User Datagram Protocol
WBA	Why-Because Analysis



## CHAPTER 1: INTRODUCTION

*"Detection is, or ought to be, an exact science and should be treated in the same cold and unemotional manner." – Sir Arthur Conan Doyle*

The history of forensics dates back to Archimedes in 287BC, who examined the principles of water displacement and, using a density and buoyancy test, was able to prove that a crown was not made of gold. It was much later, in 1822, that Francis Galton's first recorded study of fingerprints led to a new branch of science known as forensics.

Computer forensics is the name given to the science of detecting digital crimes committed in cyberspace without finite geographic locations. Dr H.B. Wolfe defined computer forensics as "A methodical series of techniques and procedures for gathering evidence, from computing equipment and various storage devices, digital media, that can be presented in a court of law in a coherent and meaningful format" [1]. Computer forensics, as defined by Wolfe, is of concern in this thesis. The focus is on the methods used to identify and detect digital crimes using computing devices and how to acquire data that can support legal criminal proceedings.

There is a considerable amount of work being done in the field of computer forensics but one of the main problems at the moment is guaranteeing the accuracy and reliability of digital evidence collected during forensic investigations. There is often a possibility of digital evidence being changed or tampered with at any stage of the analysis. This problem is called the anti-forensics [2] as it undercuts the validity of digital evidence. Due to the complexity of the subject, there is

currently no consensus on a precise definition for anti-forensics. Therefore, this thesis will draw from the best definitions so far to understand anti-forensics. Ryan Harris defines anti-forensics as “*any attempts to compromise the availability or usefulness of evidence to the forensics process.*” [3]. On a high level, Harris presents a useful first step towards what this thesis will consider as anti-forensics.

Currently, the challenge is to develop a framework for stopping anti-forensic activity. Traditionally, once evidence is collected a chain of custody is built and the evidence analysed. The evidence is presented in a court of law and may result in prosecution. In the absence of a formalised procedure which is capable of systematically detecting anti-forensic activity of evidence, these traditional methods rely purely on the skills and experience of a forensic examiner. What computer scientists have begun to realise is that without a systematic analysis of anti-forensic activity the chain of evidence custody is no longer sufficient to ensure evidence integrity. A framework is needed to identify anti-forensics. To help further the research in this field, a meta-forensic model is proposed in this thesis. By “meta-forensic”, in this thesis we mean a step above and beyond, forensic detection. The term is used to identify the crime (anti-forensics) by employing techniques like simulating the crime, identifying the method used and compare and contrast the results. The entire process is termed meta-forensics. This dissertation will not contend, and is not supporting the idea, that meta-forensics is somehow removed from computer forensics itself. Meta-forensics is not a separate field. Rather, it is integral to the computer forensics process. The term “meta-forensics” is introduced for the sake of clarity and means the forensics used to identify anti-forensics. The meta-forensic model also contributes in advancement of already existing models used in different areas of security such as:

- 1) Policy Based Security Model
- 2) Common Information Model (CIM) for Security
- 3) Vulnerability Tree Model
- 4) IDS Security Model

The proposed meta-forensic model is a framework, which addresses the anti-forensic problem as a whole and is applicable to validate any computer forensic evidence. The proposed model will be implemented using various open source security tools and Intrusion Detection System (IDS).

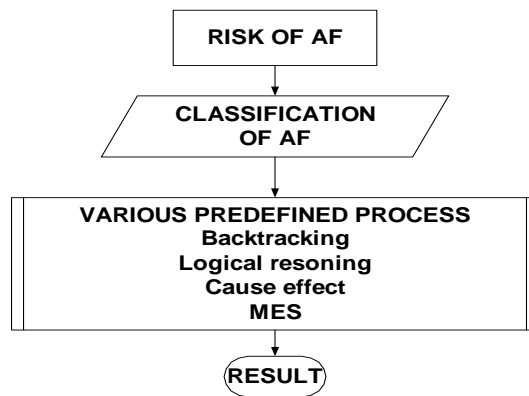
### **1.1 The Hypothesis**

This research proposes the following hypothesis:

*“There is always a risk of potential undetected anti-forensic activity in the system before and during forensic investigation process”.*

One can argue that a risk alone cannot establish anti-forensics as either AF activity is present or it's not present. The system of threat levels has always been around other areas of security, one such been created by UK home office [4] to keep us informed about the level of threat the UK faces from terrorism at any given time. This research tries to establish the likelihood of an anti-forensic activity under a given circumstance. The presence of incriminating tool or vulnerability alone will not establish anti-forensics as these tools are also used for privacy management and normal system operations. For example an enhanced system delete tool installed by user for

better system management can be classified as normal system tool or an anti-forensic tool which is debatable. A mere presence of system vulnerability will not establish anti-forensics. In this research we argue that by investigating the events surrounding the use of these tools by backtracking an incident, using various logical reasoning techniques like attack tree vulnerability tree reasoning, MES graph, why because analysis, why because graph we can establish if an anti-forensics has taken place. It's been demonstrated by way of experiment that a successful vulnerability exploit constitutes an anti-forensics.



**Figure 1: Diagrammatic representation of investigation of hypothesis**

The diagram shows the overview of how the hypothesis will be investigated. We assume that there is always a risk of undetected AF. To address this we classify the AF and run it through the predefined process which establishes the risk and likelihood of AF activity. The entire process has been named as meta-forensic model. The result will help forensic investigator to decide how to investigate a given forensic case when AF claims have been made during and before investigation.

## 1.2 The Basics of Forensic Investigation

The basic steps [2] used in a forensic investigation are:

- Acquire
- Authenticate
- Analyse

In the event of a suspected digital crime, the case is assessed initially by planning the resources and risk involved in the investigation process. The original evidence is preserved and the data recovered is used in the analysis process. Then a case report is prepared for the prosecution. All forensic investigations follow the above three steps. They are widely known as the “3 A’s” in the forensic world [2]. The proposed meta-forensic model presented in subsequent chapters defines a fourth “A”, an anti-forensic step. The fourth step works across all the steps of forensics to detect the threat of anti-forensic activity. At every stage of forensics the existence of anti – forensic activity is questioned.

## 1.3 Anti-Forensics

A widely accepted definition was coined by Dr. Marc Rogers of Purdue University. “*Attempts to negatively affect the existence, amount and/or quality of evidence from a crime scene, or make the analysis and examination of evidence difficult or impossible to conduct*” [5]. The rationale behind anti-forensics is to stop investigators finding the perpetrator or the act by contaminating the evidence. As Rogers’s points out, anti-forensics affects the evidence in a crime scene, it amounts to the digital equivalent of being able to change one’s fingerprints on a physical crime scene. The point is to avoid being caught. Scott [6] further elucidates Rogers point: “*Anti-*

*forensics is more than technology. It is an approach to criminal hacking that can be summed up like this: Make it hard for them to find you and impossible for them to prove they found you.”*

The main purpose of anti-forensics is to hide or distract from what is happening. The goals of anti-forensics are:

- Avoid Detection.
- Corrupt the information collection process or to make it look as if it's corrupted.
- Lead to false data.
- Increase the time of investigation.
- Disable detection tools.
- Destroy valuable evidence.
- Destroy the confidence in gathered evidence.

In short, anti-forensics seeks to disrupt forensic investigations.

#### **1.4 Research Aims and Objectives**

The main aims of this research are:

- To develop a standardised approach to validate forensic data taking into account the possibility of anti-forensic activity.
- To develop an AF activity monitor for network security.

Achievement of the aims proposed above depends on completion of the following Objectives:

- To provide an overview of current forensic and anti-forensic procedures using a model-based approach.

- To show the degree of anti-forensic activity on forensic data collected during an investigation.
- Investigate various security models and process with respect to forensics in a controlled environment.
- Investigate the validity of the reliable forensic evidence from the acquired data source.
- Propose a new model-based approach (which uses fuzzy logic) that can assess the level of anti-forensic behaviour.
- Determine all possible scenarios of evidence compromise by backtracking transitions from the state in which the system was discovered.
- Develop an effective meta-forensic security procedure for digital evidence validation.
- Develop a systematic logic to assign weights to anti-forensic events for digital evidence validation.
- To examine a test case in which anti-forensic behaviour was in evidence.

## **1.5 Dissertation structure**

Chapter 1 presents the motivation for the research, the background of computer security, the basics of forensics and anti-forensics, and the aims and objectives.

Chapter 2 describes the relevant concepts of digital forensics. This is done before the literature review to establish the fundamental concepts and ideas that underpin this dissertation. The chapter reviews process, evidence, proof, and the role of the forensic expert in the digital investigation process. It defines digital evidence and associates it with technology and its use. It

describes various anti-forensic techniques and fuzzy logic in the context of digital forensic investigation. It also introduces the concept of anti-forensics and various ways available to compromise digital forensic data.

Chapter 3 reviews how digital forensic data can be collected using various models by comparing and contrasting these models. It presents a critical literature review of the existing approaches for digital forensic investigation and forensic models. It discusses how the logic can be used to compare events against each other.

Chapter 4 describes the need for a validation model by drawing conclusions from the literature review and presenting an argument for a new approach. This chapter defines the logic behind the meta-forensic model identifying the problem domain and expressing it as a problem statement. Experiments performed in this research are presented that argue the need for a validation model. This chapter discusses and exposes weakness in the system which can be exploited using available tools and techniques.

Chapter 5 proposes a meta-forensic model which is implemented using a fuzzy logic approach. The model presented in this chapter addresses the issue identified in the literature review. A real world case study is discussed highlighting the benefit of verification of the digital evidence collected for a successful prosecution.



Finally, Chapter 6 gives the conclusions for the work, as well as directions for future research. The findings of the research which have the potential to add value to the current tools are presented.

## **1.6 Summary of research achievements**

The outcome of this research highlights the current methods available in computer forensics and security models. The research in this field brings together various security models involving forensic and meta-forensic approaches by formulating standards for future models. There exist various anti-forensic technologies that can destroy the evidence collected at various stages of investigation. The outcome of this work contributes to the challenge of anti-forensics and emphasises the need to validate data using meta-forensic approaches for future forensic requirements. The achievement of this research is the development of a meta-forensic model which can be used to evaluate the varying degrees of anti-forensic activity on any data collected. This research proposes a new concept based on meta-forensic principles to retrospectively look into data collected from various sources and to integrate it with existing models.

- The thesis attempts to provide guidance to manage security using a meta-forensic approach.
- The thesis proposes a new meta-forensic model to validate digital evidence.
- The thesis proposes ways to evaluate tools and software's used in forensics and anti-forensics.

- The thesis reviews current security model and investigation process and identifies the weakness in each model.
- The thesis proposes new logic for representation of anti-forensics events.

The Contribution to the knowledge in this field is discussed in chapter 6.

## CHAPTER 2: CONCEPTS OF DIGITAL FORENSICS AND ANTI-FORENSICS

This chapter will introduce the fundamental concepts of digital forensics and anti-forensics. In order to carry out a critical analysis of the literature it is necessary to establish a base from which to proceed. The structure of this chapter is as follows. In the first section, the process of collecting will be explained, detailing classes of evidence as well as information contexts. In the second section case evaluation is considered. The third section concerns reconstruction and validation. Sections four and five introduce anti-forensic concepts and methods. Cryptography will be discussed in order to explain techniques used for uncovering and understanding information that has been hidden. In the last section, fuzzy logic will be explained in order to establish a basis for the anti-forensic model recommended by this research.

Digital forensics is a branch of science dealing with digital information produced, stored, and transmitted by computers as a source of evidence in all investigations and legal proceedings. The Digital Forensic Research Workshop has defined digital forensics as “*The use of scientifically derived and proven methods toward the preservation, validation, identification, analysis, interpretation, documentation, and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorised actions shown to be disruptive to planned operations*” [7]. There are a variety of other definitions of digital forensics provided in the literature:

1. *“Computer forensics is a science that is concerned with the relation and application of computers and legal issues [8]”.*
2. *“Computer forensics science is the science of acquiring, preserving, retrieving, and presenting data that has been processed electronically and stored on computer media [9]”.*
3. *“Computer forensics is the process of methodically examining computer media for evidence [10]”.*
4. *“Computer forensics is the collection of techniques and tools used to find evidence on a computer that can be used against one in a court of law [11]”.*
5. *“Computer forensics is the study of computer technology as it relates to the law [12]”.*

It is certainly the case that concepts of forensics as defined in 1 address the relationship between application of computers and legal issues which is not sufficiently precise and helpful. The same could be said of definition 5. Definition 4 talks of techniques and tools but is felt that this misses the point, especially as it shifts the responsibility onto the tools without stipulating the logical requirements. Definition 3 is somewhat scientific with a methodical approach to definition 2 but the key issues are not covered by these two definitions, namely the tools of “acquiring, preserving, retrieving” data for a methodological approach. However, none of the definitions addresses the validity or reliability of the evidence and this thesis adopts the definition of anti-forensics in section 2.4 under anti-forensic concepts.

## 2.1 The Process

Investigative digital forensics can be divided into several stages according to the Digital Forensic Research Workshop [13] and its examination of digital forensic models. The different stages are:

- Identification: Recognising an incident from indicators and determining its type. This is not within the field of forensics, but significant because it impacts other steps and determines if a forensic examination is needed.
- Preparation: Preparing a plan of action by selecting tools, techniques, monitoring authorisations and management support. This also includes warrants if the evidence lies with a third party.
- Preservation: The preservation stage tries to freeze the crime scene. It consists of stopping or preventing any activities that can damage the digital information being collected like using electromagnetic devices, stopping ongoing file deletion processes and stopping any scheduled jobs which might interfere with the evidence.
- Collection: Collecting digital information relevant to the investigation. The evidence is duplicated in some other medium. It may involve removal of personal computers and hard disks from the crime scene, copying log files from computer devices and taking system snapshots of the devices involved.
- Examination: Examination stage consists of in-depth systematic search of evidence relating to the suspected crime. This stage focuses on identifying and locating potential evidence, within unconventional locations, and constructing detailed documentation for analysis. The outputs of examination are data objects found in collected evidence. They

may include log file time stamps matching the security camera timestamp. It is a mapping process of all the evidence collected.

- **Analysis:** The aim of analysis is to draw conclusions based on the evidence found. Different types of evidence are linked during this process.
- **Presentation:** Summarises and provides explanations of conclusions based on the analysis report. The technical data is translated into layman's terms using abstracted terminology. All abstracted terminology should reference the specific details.
- **Returning evidence:** Ensuring physical and digital property is returned to its proper owner after the investigation. It's not a forensic step but a clean way of concluding the investigation.

This is a well documented process and the Examination stage is certainly a first step towards a meta-forensic approach. The Examination phase is intended to check the validity of the evidence collected and corroborate the data location, chain of custody and how the information was acquired versus any unforeseen data hiding or pitfalls. However, anti-forensic activity goes one step beyond what the Examination phase checks for, as attempts would have been made to corrupt evidence even though it has passed Examination. Thus, a further step is required during the collection and examination phase to establish the validity of evidence depending on AF threat level.

### **2.1.1 Evidence**

There are two basic types of evidence, physical evidence and digital evidence. Physical evidence refers to tangible items that "*furnish or tend to furnish proof*" [14]. Digital evidence also

furnishes or tends to furnish proof and is admissible in court of law, so this research considers all the evidence as tangible. Digital evidence is electronic in nature and can be found as data on computer systems that could refer to documents or events that occur within a computer system or network. It includes files stored on computer hard drive, digital video, digital audio, network packets transmitted over local area network, wide area network, e-mails, browsing history, databases etc.

The evidence can be

- Direct evidence: In digital evidence if an illegal image is found in a computer the seized computer becomes evidence.
- Circumstantial evidence: This is also known as indirect evidence. Circumstantial evidence relates to a series of facts other than the particular fact sought to be proved. The existence of specific hacking tools or malware on a suspect's computer is important circumstantial evidence.
- Evidence of Intent: What was person's intention is usually a matter to be determined by inference. In digital evidence an online bank transfers at a given time can show monetary benefits as an evidence of intent.

When collecting evidence the following must be ensured:

- The evidence is authentic, or it relates to the incident;
- The evidence is complete;
- The evidence is reliable;
- The evidence is believable.

Vacca's [15] above procedure does not consider any AF activity threat and this thesis will try to establish a procedure to adhere to in this research, however it is worth commenting on Gödel's incompleteness theorem in this respect. What Gödel proved is that in an effective procedure, like this process for collecting evidence, there will always be statements that are un-provable by the system even though they are true. An evidence statement E which states "there is no proof of E". If E is true, there is no proof of it. If E is false, there is a proof that E is true, which is a contradiction. Let us consider evidence of intent E= there is no proof that it's the same person who performed the online transaction committed AF activity. So there is no proof of evidence statement. Therefore it cannot be determined within the system whether E is true. Thus, the evidence collected is left to the interpretation of the forensic examiner as its either complete or incomplete but cannot be both. This concept is used in section 4.3 and 5.3 for perception based information.

### **2.1.2 Classes of Digital Evidence**

Depending on the nature of the case, and the facts to be proved, digital evidence falls into different classes of evidence.

- Possession of certain hardware such as key loggers.
- Digital images or software presented.
- E-mail messages presented as proof of their content.
- Access log files with time stamps of system information such as access log files.



- Digital signature technology like PGP and private key encryption can be proved as paper based document testimony. This depends on a company policy of using digital processes rather than paper-based process.

### **2.1.3 Anonymity of digital information**

Digital information documentation generated, stored, and transmitted between electronic devices never bears any physical imprints connecting it to the individual who caused its generation. Unless the information is recorded by an external source with voice recognition software, finger print or a photographic system there is nothing intrinsic linking digits to a person. The individual logins to computers do not identify the users generating specific digital data nor does a token login associate the person with the digital data being generated. The current logging software logs and assigns identity to the data at an user level but there is no way to verify its origin without initiating a full forensic investigation [16]. This anonymity of digital information plays a key role in anti-forensics as users and forensic investigator make claims and counter claims on who actually generated the data. So the possibility of anti-forensic activity always exists.

### **2.1.4 Standard and burden of proof**

The “standard of proof” is the level of proof required in a legal action to discharge the burden of proof, i.e. convince the court that a given proposition is true [17]. There are generally three broad types of burdens.

- A “legal burden” or a “burden of persuasion” is an obligation that remains on a single party for the duration of the claim.
- An “evidentiary burden” or “burden of leading evidence” is an obligation that shifts between parties over the course of the hearing or trial. A party may submit evidence that the court will consider prima facie proof of some state of affairs. This creates an evidentiary burden upon the opposing party to present evidence to refute the presumption.
- A “tactical burden” is an obligation similar to an evidentiary burden. Presented with certain evidence, the Court has the discretion to infer a fact from it unless the opposing party can present evidence to the contrary.

### **2.1.5 Context of digital information**

The context in which the digital information is referred depends on how it was generated. If third party devices or computer programs produce the information, it follows a file system format such as NTFS and the FAT file system will be used. The format prescribes how the information is to be interpreted. If the information is produced for internal use by a proprietary computer program, there are usually no public standards available to describe and interpret the digital information. If this is the case, the investigator must understand the system device or program from which the evidence is gathered. Before considering digital evidence the context determining the meaning of the information must be clarified.

### **2.1.6 Legal Framework**

Legal issues are very important as violation of legal commitments damages the reputation on an organisation. Computer crime laws which is also known as cyber laws deals with unauthorised access, modification or deletion, disclosure of sensitive information, use of unauthorized software's in computer and network.

Following are some of the laws [18, 19] that are created or modified in United States and United Kingdom to cover various types of computer crimes:

- US Computer Fraud and Abuse Act of 1986
- US Electronic Communications Privacy act 1986
- US Computer Security act 1987
- US Child Pornography Prevention Act of 1996
- US PATRIOT Act of 2001
- US Sarbanes-Oxley Act of 2002
- US CAN-SPAM Act of 2003
- The Council of Europe's Convention on Cybercrime of 2001 (Europe)
- The Computer Misuse Act of 1990 (UK)
- The Data Protection Act 1998 (UK)

In the United Kingdom examiners usually follow guidelines issued by the Association of Chief Police Officers (ACPO) for the authentication and integrity of evidence [20]. The guidelines consist of four principles:

- No action taken by law enforcement agencies should change data held on a computer or storage media which may subsequently be relied upon in court.
- 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.
- An audit trail or other record of all processes applied to computer based electronic evidence should be created and preserved. Third party should be able to examine those processes and achieve the same result at any time.
- The officer in charge of the investigation has overall responsibility for ensuring that the law and these principles are followed.

These guidelines are widely accepted in courts of England and Scotland, but they do not constitute a legal requirement and their use is voluntary.

## **2.2 Evaluating the case**

The case can be assessed in the following manner:

- Situation of the case: This depends on the circumstances which led to forensic investigation.
- Nature of the case: The case can be a breach of company policy or intended to cause criminal damage to others.
- Types of evidence: The evidence, which links the person to the crime, can be both physical and digital. There can be paper-based evidence linking it to digital evidence.
- Technology used by suspect: Type of operating system and hardware used.

- Infrastructure: Description of how the computer is connected to the network. This helps the investigator to locate the evidence he or she is looking for.
- Location of Evidence: The physical location of the evidence and the access to it. The evidence can be distributed across various geographic locations.
- The motive: Understanding what caused the crime and the motive behind it helps to locate the evidence.
- Warrant: To carry out an investigation a search warrant from a court is required. The warrant can be for an entire company or just a device inside a building.
- Documentation: Every finding needs to be documented including hardware configuration, system date and time, document file names, last logon, account usage.

All the above procedure follows a standard practice without being validated for any anti – forensic activity. It can be argued that every step of the evaluation is prone to anti-forensics.

- Situation of the case: The circumstances can change during course of investigation. Was a circumstance created by someone?
- Types of evidence: Was the evidence type classified correctly?
- Technology used by suspect: Was the technology identified correctly?
- Infrastructure: Was the Infrastructure changed on or before or during investigation?
- Location of Evidence: Evidence can be distributed across various geographic locations. Was it moved?
- Warrant: Was a search warrant issued to cover everything? Was any device left out?

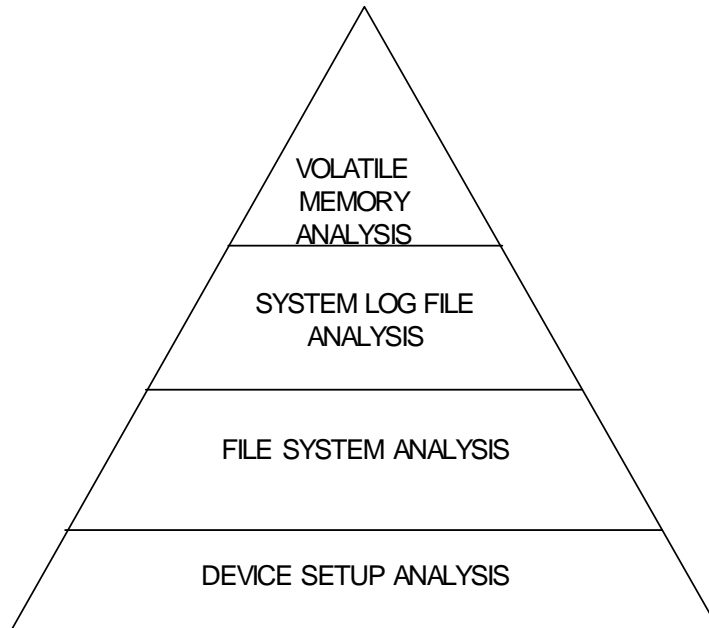
## **2.3 Reconstruction and Validation**

The possession of an incriminating information file from the compromised system does not prove that the owner of the device is responsible for the objects in it. Apart from the owner of the device, the information can be generated automatically by a rogue system. The information could have been planted via Trojan, virus, and spyware or it may have been left by the previous owner of the computer. To determine responsibility the investigator must validate the evidence in their possession by any one of the methods proposed later in this thesis. Reconstruction of events can be used to validate the evidence collected, however the investigator has to be familiar with the device, the local network, and systems interconnectivity.

This thesis classifies reconstruction for anti-forensics validation according to the primary object of analysis from the infrastructure levels. It can be represented as a pyramid structure. Major classes identified are:

- Device setup analysis.
- File system analysis.
- System log file analysis.
- Volatile memory analysis.

The classes are represented as a pyramid structure to illustrate the fact that forensic investigator needs to look into the infrastructure level before moving on to higher level to gather evidence. We represented this as a pyramid because the initial device setup requires a broader approach base which slowly converges to a narrow area when we move upwards.



**Figure 2: Reconstruction and validation of primary object of analysis**

### **2.3.1 Device setup analysis**

For a successful forensic validation an understanding of network device setup is essential. The device being investigated may be linked with a specific make of router. The possible types of network topology are discussed in Appendix A. A possibility of a rouge device being introduced in device setup or incorrectly configured device can produce results which can also result in an anti-forensic activity. The experiment in Chapter 4.4 considers one such possibility.

### **2.3.2 File system analysis**

There are different types of file system. File system types can be classified into disk file systems, network file systems and special purpose file systems. The possibility of anti forensic activity in these systems depends on the exploited vulnerability.

## Disk file systems

A disk file system is a file system designed for the storage of files on a data storage device, most commonly a disk drive, which might be directly or indirectly connected to the computer. Examples of disk file systems include FAT, FAT32, NTFS, HFS and HFS+, ext2, ext3, ISO 9660, ODS-5, and UDF. Some disk file systems are journaling file systems or versioning file systems. There are several ways to duplicate file system information. The method available depends on circumstances. The investigator captures information by logging into a compromised machine, listing files on the terminal, and recording the session with a terminal emulator program.

- *“Copying individual files. This is the least accurate approach, because it captures only the content of files”*. No meta-information is captured except perhaps the file size (however, holes in files become indistinguishable from zero-filled blocks, increasing the apparent file size. All other meta-information such as file ownership, access times, and permissions is lost unless it is saved via some other means. A vital part of the anti – forensic activity may be missed by this approach.
- Making a backup. Depending on the backup software used, this preserves some meta-information such as ownership, information about hard links, and last modification time, but it does not capture the last read access time. Commonly used UNIX utilities are tar, cp, or dump. The drawback of making a backup is that what you see is all you get. Sometimes using UNIX commands like cp-r -p or rsync can preserve permissions of files. Backups do not capture information about deleted files missing a possible anti-forensic activity.



- Copying individual disk partitions. This creates a bit-for-bit identical copy of each file system, including all the meta-information, and including all the information that sits in unallocated space at the end of files, between files, in unallocated inode blocks, and so on. This is typically done with the “dd” command which is available in operating system. A major benefit of this approach is that it is file system neutral. The downside is that one still misses data that is stored between and outside partitions with possible AF evidence.
- Copying the entire disk. This time, the result is a bit-for-bit identical copy of all accessible information on the disk, including storage space before or after disk partitions. This can be necessary when suspicion exists that data could be hidden outside disk partitions. Again, “dd” is the preferred command for doing this. Even this method has limitations: it will not read disk blocks that have developed errors, and that the hardware has silently re-mapped to so-called spare blocks, nor will this method give access to unused spare blocks, as they lie outside the normally accessible area of the disk.

The uncertainty of the captured information increases the dependency on the integrity of the compromised system. For example, when individual files are captured while logged into the victim machine, a subverted application or kernel software could distort the result. The lack of accuracy could possibly miss an anti-forensic activity.

### **Network file systems**

The Network File System is an open standard defined in RFCs, [21] allowing anyone to implement the protocol. Few of the examples of network file systems include NFS, SMB protocols.

### 2.3.3 System log file analysis

A log file is a file that lists actions that have occurred. All devices maintain log files listing every request made to the server. A log file entry usually consists of a timestamp, IP address, process identifier that generated the entry, some technology description like a browser used to access a website and the reason for generating an entry. The log files are generated for system usage, user interaction and to maintain system health. The knowledge of circumstances in which processes generate log file entries permits forensic scientists to infer from presence or absence of log file entries that certain events have happened. For example in a web server, running IIS generates multiple entries in log files when an URL based double dot attack is attempted on a vulnerable system. The example [22] presents an old, well-known, vulnerability found on IIS versions 4.0 and 5.0, where an attacker could bypass authorisation schema and gain access to any file on the same drive as the web root directory due an issue on decoding mechanism. This is possible when an attacker could execute arbitrary commands on the web server by submitting URL's :

Original URL:

```
http://victim/finance site/../../../../winnt/system32/cmd.exe?/c+dir+c:\
```

Double encoded attack URL:

```
http://victim/finance/site/%252E%252E%252F%252E%252E%252Fwinnt/system32/cmd.exe?/c+dir+c:\ "
```

Attacks like this emphasis the fact of reconstruction and validation of forensic evidence however without a systematic procedure to identify these kinds of attacks the AF activity can go unnoticed. One of the components of the proposed model is to address this issue.

### 2.3.4 Volatile memory analysis

If the system under investigation is secured without a restart or shutdown volatile memory (e.g. RAM) can be analysed by checking the task manager in windows. This will list the processes in memory. For a UNIX system, the top command will list the processes. A sample of the listed task is shown below. To analyse the complex process threads the operating system providers provide tools to list the dependencies .One such tool is Process Explorer v11.04.

Volatile memory analysis determines the current state of the system including Date and time, open ports, running processes, port and processes mappings. As this information is volatile and keeps changing a dump of this information is useful for later analysis. Volatile memory analysis looks for some basic information during or after a system compromise. The listing shows the process along with memory and CPU information. This is useful to determine and isolate genuine system process from the rogue process. In Fig5 the top command lists the operating system process and port scan map which is active. The snapshot of this process is one of the steps in analysing the system.

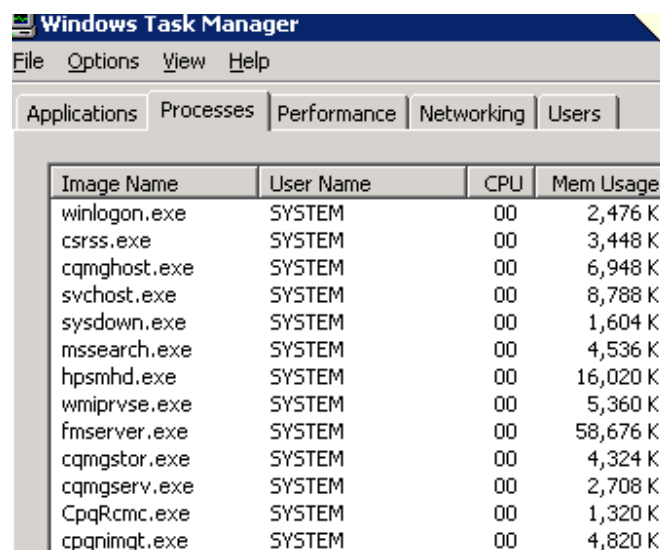


Image Name	User Name	CPU	Mem Usage
winlogon.exe	SYSTEM	00	2,476 K
csrss.exe	SYSTEM	00	3,448 K
csrss.exe	SYSTEM	00	6,948 K
svchost.exe	SYSTEM	00	8,788 K
sysdown.exe	SYSTEM	00	1,604 K
mssearch.exe	SYSTEM	00	4,536 K
hpsmhd.exe	SYSTEM	00	16,020 K
wmiprvse.exe	SYSTEM	00	5,360 K
fmserver.exe	SYSTEM	00	58,676 K
csrss.exe	SYSTEM	00	4,324 K
csrss.exe	SYSTEM	00	2,708 K
CpqRcmc.exe	SYSTEM	00	1,320 K
cpqimngt.exe	SYSTEM	00	4,820 K

Figure 3: List of process and memory usage in windows

```

cpu02 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 100.0%
cpu03 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 100.0%
Mem: 8139028k av, 2064076k used, 6074952k free, 0k shrd, 47744k buff
     1713284k active, 195300k inactive
Swap: 16776952k av, 0k used, 16776952k free 495952k cached

```

PID	USER	PRI	NI	SIZE	RSS	SHARE	STAT	%CPU	%MEM	TIME	CPU	COMMAND
5097	root	21	0	1290M	1.3G	118M	S	4.9	16.2	60:37	3	java
20551	root	20	0	1176	1176	872	R	0.9	0.0	0:00	0	top
1	root	15	0	500	500	440	S	0.0	0.0	0:05	3	init
2	root	RT	0	0	0	0	SW	0.0	0.0	0:00	0	migration/0
3	root	RT	0	0	0	0	SW	0.0	0.0	0:00	1	migration/1
4	root	RT	0	0	0	0	SW	0.0	0.0	0:00	2	migration/2
5	root	RT	0	0	0	0	SW	0.0	0.0	0:00	3	migration/3
6	root	15	0	0	0	0	SW	0.0	0.0	0:00	2	keventd
7	root	34	19	0	0	0	SWN	0.0	0.0	0:00	0	ksoftirqd/0
8	root	34	19	0	0	0	SWN	0.0	0.0	0:00	1	ksoftirqd/1
9	root	34	19	0	0	0	SWN	0.0	0.0	0:00	2	ksoftirqd/2
10	root	34	19	0	0	0	SWN	0.0	0.0	0:00	3	ksoftirqd/3
13	root	25	0	0	0	0	SW	0.0	0.0	0:00	1	bdflush
11	root	15	0	0	0	0	SW	0.0	0.0	0:00	1	kswapd
12	root	15	0	0	0	0	SW	0.0	0.0	0:00	1	kscand
14	root	15	0	0	0	0	SW	0.0	0.0	0:00	1	kupdated
15	root	25	0	0	0	0	SW	0.0	0.0	0:00	0	mircoweryd
23	root	15	0	0	0	0	SW	0.0	0.0	0:00	0	kjournald
81	root	25	0	0	0	0	SW	0.0	0.0	0:00	1	khubd
261	root	22	0	0	0	0	SW	0.0	0.0	0:00	3	kjournald
262	root	15	0	0	0	0	SW	0.0	0.0	0:00	3	kjournald
922	root	15	0	620	620	532	S	0.0	0.0	0:00	3	syslogd
926	root	15	0	464	464	404	S	0.0	0.0	0:00	3	klogd
937	root	15	0	456	456	392	S	0.0	0.0	0:00	2	irqbalance
955	rpc	17	0	576	576	504	S	0.0	0.0	0:00	0	portmap
975	rpcuser	25	0	712	712	636	S	0.0	0.0	0:00	0	rpc.statd
1016	root	15	0	5868	5868	2372	S	0.0	0.0	0:01	0	smmpd
1045	root	15	0	3132	3132	1544	S	0.0	0.0	0:00	0	cupsd
1073	root	15	0	1560	1560	1308	S	0.0	0.0	0:00	1	sshd
1089	root	23	0	852	852	728	S	0.0	0.0	0:00	3	xinetd
1103	ntp	15	0	2560	2560	2196	S	0.0	0.0	0:00	3	ntpd
1113	root	25	0	396	396	336	S	0.0	0.0	0:00	0	rpc.rstatd
1134	root	15	0	2548	2548	1876	S	0.0	0.0	0:00	3	sendmail
1143	smmmp	15	0	2292	2284	1748	S	0.0	0.0	0:00	0	sendmail
1177	root	15	0	476	476	420	S	0.0	0.0	0:00	0	gpm
1360	root	15	0	6488	6488	3228	S	0.0	0.0	0:00	1	hpsmhd
1361	root	25	0	900	900	696	S	0.0	0.0	0:00	3	rotatelogs
1362	root	25	0	892	892	688	S	0.0	0.0	0:00	0	rotatelogs
1373	root	15	0	1016	1016	648	S	0.0	0.0	0:00	3	crond
1439	xfs	15	0	4932	4932	956	S	0.0	0.0	0:00	3	xfs
1603	root	19	0	1168	1040	740	S	0.0	0.0	1:54	1	hpsmhd

Figure 4: List of process and memory usage in UNIX

## 2.4 Anti-forensic concepts

Anti-forensic activity existed before it was formally defined. A definition of anti-forensics was discussed in section 1.3 but none of it considered the judicial review. An apt definition was coined by Liu and Brown [23], as the, “*application of the scientific method to digital media in order to invalidate factual information for judicial review.*” Meta-forensics uses the techniques used by anti-forensic activity in order to develop security tools for privacy. In other words, as Liu and Brown state that anti-forensic activity explicitly is used to invalidate factual information for judicial review, it can’t ever be used for other purposes. Meta-forensics, on the other hand

adopts and learns from the processes used by anti-forensics for other purposes, namely to identify anti-forensics. It depends on individual use and approach to anti-forensics and through this research we try using meta-forensics for constructive purposes.

## **2.5 Anti-forensic methods**

Rogers [24] classified methods used in anti-forensics into four basic categories:

- Data hiding
- Artifact wiping
- Trail obfuscation
- Attacks against the computer forensics process or tools

One can argue that the classification is not necessary as categories can always overlap. However to detect and demonstrate a possibility of AF activity classification helps us to identify more accurately the inter relationship between each anti-forensic activity and events. The inter relationship of a particular event is easy to identify and visualize on a classified category. The categories will form part of the framework to develop a forensic language for identifying AF activity in system discussed in Chapter 5.

### **2.5.1 Data hiding**

The word steganography is of Greek origin and means “concealed writing”. The first recorded use was 500 years ago in 1499 by Johannes Trithemius [25] in his *Steganographia*, a treatise on cryptography and steganography disguised as a book on magic. Steganography is the technique

of hiding a message or a file, usually by making the file appear to be something else. A well know practice in anti-forensic activities is to hide files as operating system files under the operating system tree structure to avoid detection. They can also be transmitted from one source to another source electronically, hidden inside an audio file, document file, image file, program or protocol. Data can be hidden in the slack and unallocated spaces on computer hard drives, as well as the metadata of many types of files. A concealed message by tampered executable files is one other way of hiding data. Various methods can also be employed to hinder computer forensic investigation. For example, a person can hide a map of an airport in a picture, table, or text block under a public blog site without being detected. Alternatively, a white text block over a white background can store a hidden message. Morse code messages can be embedded in a picture. Null ciphers form messages by selecting a pre-determined pattern of letters from a sequence of words. Many other forms can be employed which makes it difficult for automated tools to detect data hiding [26].

### **2.5.2 Artifact wiping**

Artifact wiping involves the deletion of particular files or entire file systems. This can be accomplished through the use of a variety of methods that include disk cleaning utilities, file wiping utilities and disk destruction techniques [27]. Disk cleaning utilities use different methods to overwrite the existing data on disks. The current DOD [28] policy states that only certain acceptable standards of wiping can clean data effectively. Disk cleaning utilities leave

their own signatures revealing that the file system was wiped. Some of the widely used disk cleaning utilities includes DBAN, SRM, [29]. Another option which is approved by the NIST (National Bureau of Standards) and the NSA (National Security Agency) is CMRR Secure Erase, which uses the Secure Erase command. File wiping utilities are used to delete individual files from a disk. They are small light weight programs mostly built into the operating system for disk sanitation. The advantage of this file deletion is it can be accomplished in a very short amount of time. They require user involvement in the process and don't always correctly and completely wipe file information [27]. Some of the widely used file wiping utilities includes R-Wipe & Clean, Eraser, B delete. Disk degaussing/destruction techniques is a process by which a magnetic field is applied to a digital media device which result in a device that is entirely clean of any previously stored data. Degaussing is an expensive technique and needs specialised equipment. A more commonly used technique to ensure data wiping is the physical destruction of the device. The NIST recommends that “physical destruction can be accomplished using a variety of methods, including disintegration, incineration, pulverising, shredding and melting” [30]. Degaussing is rarely used as an anti-forensic method as it involves use of specialised equipment but presence of wiping tools can increase the possibility of AF activity.

### **2.5.3 Trail obfuscation**

Trail obfuscation is to mislead and divert the forensic examination process. It is conventionally explained that trail obfuscation covers a variety of techniques and tools that include “log cleaners, spoofing, misinformation, backbone hopping, zombie accounts, trojan commands”[31]. Trail obfuscation can also be accomplished by wiping and/or altering server log files and/or

system event files, or altering the dates of various files using touch, rename, Timestamp which is part of the metasploit framework. Another well known trail-obfuscation program is Transmogrify (also part of the metasploit framework). Transmogrify allows the user to change the header information of a file, so a (.jpg) header could be changed to a (.doc) header. When a forensic tool is used to conduct a search for images on a machine, it would simply see a (.doc) file and therefore skip this file [32]. Although the tool looks for an image file but failed to find it shows successful AF activity.

#### **2.5.4 Attacks against the computer forensics process or tools**

During a typical digital forensic examination, the examiner creates an image of the computers disks. This keeps the original computer (evidence) from being tainted by forensic tools. A disk hash is created by the forensic examination software to verify the integrity of the image. A recent anti-forensic technique targets the integrity of the hash that is created to verify the image. By affecting the integrity of the hash, any evidence that is collected is questionable [27]. Forensic tools, like EnCase, FTK, iLook, SleuthKit, and WinHex are all prone to attack. Custom scripts that change FAT, NTFS, and ext inodes have been around for years and they “tamper programs and write to file slack, alter file signatures, and flip bits in order to evade hashset detection” [27]. A report from the 2007 U.S. Black Hat conference showed exploitation techniques to a number of commercial and open-source computer forensics application Software [33, 34]. The report concluded with following findings:

- Forensic tools developers never took into consideration attacks against stack overflows, memory management, and exception handling leakage.



- Users never test their products rigorously before deploying it in live cases. In fact, most computer forensic labs purchase the software that “everyone else” is using and do not perform independent tests of reliability and thoroughness, particularly as new versions of the software get released.
- Forensic software users do not apply sufficiently strong criteria to the evaluation of the products that they purchase.

The work focuses on identifying AF activity with help of tools but the tools used in the process itself is questioned for reliability. The proposed model in Chapter 5.3 tries to address the issues raised here.

## **2.6 Metasploit Project**

The Metasploit project is described as [35] is a “*development platform for creating security tools and exploits. The framework is used by network security professionals to perform penetration tests, system administrators to verify patch installations, product vendors to perform regression testing, and security researchers world-wide. The framework is written in the Ruby programming language and includes components written in C and assembler.*” The Metasploit Framework is a tool for developing and executing exploit code against a remote target machine. Other metasploit projects include the Opcode Database, shellcode archive, and security research. The Opcode Database is a resource for authors who write new exploits. Buffer overflow exploits on Windows require advanced knowledge of Opcodes and DLLs if an automated tool kit is not used for attack. These differ in the various versions and patch-levels of a given operating system, and they are all documented and conveniently searchable in the Opcode Database. This allows one to

write buffer overflow exploits which work across different versions of the target operating system. The Shellcode database contains the payloads used by the Metasploit Framework. These are written in assembly language and full source code is available.

The outcomes of the Metasploit project have been the Metasploit framework which contains a suite of programs that includes:

- Sam Juicer – Which acquires the hashes from the NT Security Access Manager (SAM) files without changing anything in hard disk.
- Slacker – Hides files within the slack space of the NT file system (NTFS).
- Time stomp – Alters all four NTFS file times: modified, access, creation, and file entry update.

The Metasploit Project is also well known for evasion tools, some of which are built into the Metasploit Framework. These tools demonstrate practical methods for invalidating digital evidence collected by a forensic examiner. The project is module focused and any new tools can be easily integrated.

## **2.7 Anti-forensic Initial compromise**

Initial compromise known as zero day attack happens mainly due to a security vulnerability which has been recently discovered, targeted, and exploited by wider community. Most security-related incidents occur due to a lack of effective information technology governance and management within an organisation [36]. However, even with good management practises and

policy enforcement, break down in technical implementation can result in servers, workstations, and other network devices remaining un-patched after a new patch release and to application downtime and various other problems. This issue is compounded if management policies do not exist and responsibilities not clearly defined or are not enforced.

## **2.8 Fuzzy Logic**

It underpins much of the theoretical rationale behind the meta-forensic model this research will ultimately recommend. It is perhaps tangential to the preceding examination. However, it does tie in with the understanding of meta-forensics used in this research. It is fundamental to the adopted approach and so time now will be spent with some brief explanations and definitions. Fuzzy logic is a promising way to represent non-traditional policies, like privacy, integrity and availability. Its approach to comparing and contrasting overlapping data fits better with meta-forensic validation. Fuzzy logic, developed by Lotfi Zadeh [37], is a form of multi-valued logic derived from fuzzy set theory to deal with reasoning that is approximate rather than precise. Multi-valued logic assumes more than two truth values like existence of AF activity or potential existence of AF activity. The values that are measured (fixed IP address) and perceived are human and vague, and it is difficult to represent them formally, although that is what we require in the meta-forensic validation model. With multiple truth values, the problem becomes even harder – *“The crux of the problem, really, is the excessively wide gap between the precision of classical logic and the imprecision of the real world”* [38]. Fuzzy logic, with its ability to handle vagueness, may be better adapted to handle integrity of the data security. This has already been demonstrated by the National Computer Security Centre: [39] in the following five areas;

- Reasonable assurance.
- Separation of duty exists.
- Supportive attitude exists.
- System is efficient.
- Individual access is allowed.

The above process can be classified into fuzzy rather than crisp sets for easy representation. Fuzzy classification allows an infinite number of levels, usually in the interval between 0 and 1 which is used in meta-forensic validation. We will look at some aspects to see if fuzzy logic has the capacity to deal with vague data. We discuss a security incident to analyse capturing the data using fuzzy logic:

**INCIDENT:** Engineering web server got compromised by attackers this morning in server room 11 am on May 5 2009

Prior to Forensic investigation:

- How sure can we be that the fact above is true and accurate?
- What is the source of this fact?
- Who provided the source information?
- Who confirmed the incident took place?
- Were the date, time, and server room logs looked at to be sure the incident happened?

During Forensic Investigation:

- Was any change made to system after the server compromise?

- When and by whom were the changes made if any?
- Was any data lost?
- Was the data backed up?
- Where are the duplicate records kept?
- Was the analysed data confirmed by the system administrator?

### **Representation of Incident as Linguistic Variable**

Incident Aspect	Assurance	Rating
Source	Initials incident date of compromise	High
History	How, When, What Where incident happened	Medium
Backup	Is data recoverable from Backup	Medium
Hold backs	Date, Time, Incident is valid and/or real	High
Data corruption	Prior to Forensic investigation	High
Data corruption	During Forensic investigation	Medium
Data corruption	After Forensic investigation	Low

The incident can be represented as a table which keeps track of the data for the incident presented. This is done using fuzzy linguistic variable. Linguistic variables represent crisp information in a form and precision appropriate for the problem. The linguistic variables like “low”, “medium”, “high”, so common in everyday speech, convey information about our incident or an object under observation. The vagueness of the information is represented as a linguistic variable expressed as fuzzy ratings in terms of peoples understanding, making it easier to enter, analyse, and maintain the overall logic of the incident using fuzzy logic concepts.

Incident aspect, assurance and ratings are classified but how we arrive at this result is the advancement of this research. Kandel and Kacprzyk [40] applied fuzzy set theory to analysis security policy in information systems. This type of logic is used in chapter 5 to classify the vulnerabilities for the meta-forensic model.

## **2.9 Summary**

This chapter introduces the concepts of forensics, anti-forensics and meta-forensics. The aim was to provide the foundation for the experiment to follow in Chapter 4. The process, evidence collection and the device setup guide us towards better understanding of AF activity and set the stage for experiment to demonstrate how the AF threat is exploited. The analysis of anti-forensic concepts exposes the weakness in the security and the possibility of compromised data ending up as a final product in an investigation. The fuzzy logic concepts help us to question the evidence by comparing and contrasting. This chapter established the four main anti-forensic categories that will be used throughout this dissertation – data hiding, artefact wiping, trail obfuscation and attacks against computer forensics and tools. These four categories should be seen in the context of the conventional methods used to undermine security, such as initial compromise and the deception of security personnel. The nature of evidence in any system was also discussed where completeness and consistency can never both be achieved. The next chapter will critically review the literature.

### **CHAPTER 3: LITERATURE REVIEW OF MODELS AND PROCESS**

The previous chapter established the nature of digital forensics, anti-forensics and computer security. Computer security and digital forensics are based on models. Kurtz and Ronald [41] define a computer security model as a scheme for specifying and enforcing security policies. In this research we examine models with relevance to AF. The model may be founded upon a formal model of access rights, a model of computation, a model of distributed computing, or no particular theoretical grounding at all. Models define security and forensics, and when they are referred to it is conventionally the case that a type of model is being discussed. Model driven development [42] reduces system development time and improves the quality of the resulting products. Recent investigations [43, 44, 45, 46] have shown that security can be integrated into system-design models and that the resulting security-design models can be used to generate security infrastructure systems. When the models have a formal semantics, reasoning is possible and one can query their properties and understand the potentially subtle difference between them. This subtle difference tries to address the hypothesis of this research which detects the AF activity on given system. The formal semantic and reasoning is also use to differentiate between tools used for AF and similar tools used for privacy and prevention of AF activity. Thus, in this chapter, in order to review the literature for this research, security models, forensic models and forensic process will be analysed, because a successful exploit in any one of it can result in AF. Appraisal of the model and process is suggested. In previous work [47], security modelling language, called Secure UML was used to formalize and check non-trivial security properties. This was the closet research to our current research to validate security design models. No

research has so far looked at security models and process in light of anti-forensics.

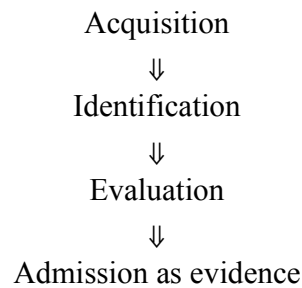
### **3.1 Modelling security and forensic behaviour**

The aim is to discuss the state of the art research in light of the need for and meta-forensics. Many of the digital forensic models engage with highly complex and specialised functions. This review considers the digital models in light of the need for preventing anti-forensic activity. The concern here is to establish where the field stands in terms of the science of forensic investigation and understanding of anti-forensics. The concern here is not with the methods used to detect the digital equivalent of a physical fingerprint, rather the concern is with the digital version of methods used to alter this finger print before, or during forensic investigation. Modelling security, forensic and anti-forensic is an interlinked process as breach of security leads to forensic investigation which may or may not detect anti-forensics. Vulnerability in any of the models can lead to anti-forensics and to address this problem forensic model is first reviewed followed by security models leading to anti forensics. Finally various reasoning approaches are reviewed in light of AF which develops as a concept for our meta-forensic model.

### **3.2 Approach to Evidence in Cyberspace**

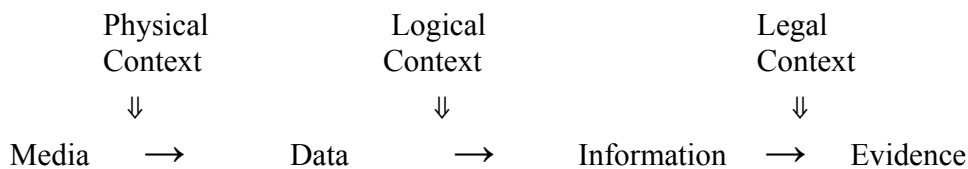
Pollit [48] in 1995 compared and mapped computer evidence and translated it into documentary evidence in a court of law. During those days of legal history, digital evidence was a new concept so it was necessary to have the testimony of someone who could explain the process of acquisition, identification, and evaluation. The process is summarised as follows:





**Figure 5: Evidence Process**

Each of these steps requires technical skills and may require testimony at trial. At this stage the media is translated into data, data to information and information to evidence. The path that digital evidence takes can be depicted as follows:



**Figure 6: Forensic Evidence Path**

Here we see the fundamentals of providing computer evidence. The challenge was to accurately transfer evidence from a physical context to a legal context. Testimony by both the forensic examiner who processed the evidence and someone who can explain its significance to the case is often required. Only then does the information become evidence. It's clear that during Pollits time technical skills and legal expertise must be combined in order to discover, develop and

utilize digital evidence. The process used must conform to both the law and science. Failure in either arena renders the product legally worthless. Ultimately, this is the aim of most data collection methods, to convert information obtained into a form that can legally be submitted as evidence. Present day anti-forensic activity, while attacking much more highly evolved models like DFRWS, still fundamentally seeks to undermine one of these key stages in the formation of evidence. Data hiding, artifact wiping, trail obfuscation and direct attacks on the forensic process all seek to fundamentally invalidate one of these steps.

### 3.3 Three-Level Hierarchical Model

Pollit [48] essentially was adapting traditional methods of forensics to computer methods. New adaptations were called for with the growth of technology and as the difficulty of solving digital crimes increased. One new model was the three level hierarchy model. This model is represented in Figure 7 below.

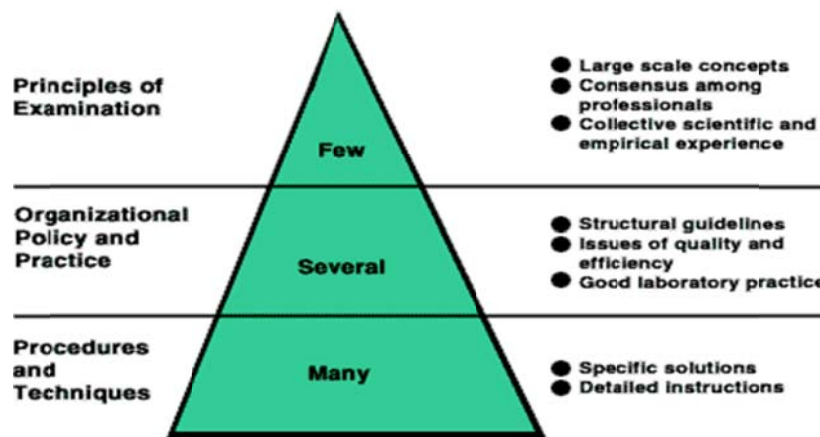


Figure 7: A Three-Level Hierarchical Model for Developing Guidelines for Computer Forensic Evidence (reproduced) [48]

The three-level hierarchical model consists of the following [48]:

- Principles of examination
- Policies and practices
- Procedures and techniques

Principles of examinations were based on collective scientific knowledge and the experience of investigators. Policies and practices were guidelines applied to forensic examinations. These were designed to ensure quality and efficiency in the workplace. These were the good laboratory practices by which examinations were planned, performed, monitored, recorded, and reported to ensure the quality and integrity of the work product. Procedures and techniques are software and hardware solutions to specific forensic problems. The procedures and techniques are detailed instructions for specific software packages as well as step-by-step instructions that describe the entire examination procedure.

As Figure 7 illustrates [48], a principle may spawn more than one policy, and those policies can accept many different techniques. It may not be the same path the examiner takes with the next case. Traditional forensic examinations, such as fingerprint and DNA examination recovered from a crime scene, lend themselves to a routine and standardised series of steps that can be repeated in case after case. There is generally no such thing as generic computer evidence procedures. The evidence is likely to be significantly different every time a digital forensic is involved. It requires an examination plan tailored to that particular evidence and this model does not standardise the way the plan is tailored. For instance an email attack on system requires high level information to identify certain types of logs and to verify if the logs have been altered by

the attacker. With this difference we still attempt to produce a model because a new type of data hiding or vulnerability may surface in future and the model can still classify it under one of the categories discussed in section 2.5 based on its properties.

### **3.4 First Research Road Map (DFRWS)**

The next step in the evolution of computer forensics happened in 2001 when the first Digital Forensic Research Workshop was held in New York [48]. So far we have looked at two models in development of computer forensics. The shortcoming with both of them is that they attempted to apply traditional forensic methods which lacked detailed steps. In retrospect this did not provide the rigour or accuracy required due to the complex environment under which computer evidence needs to be obtained. Four important considerations emerged from the DFRWS:

1. Define a Framework for Digital Forensic Science
2. Discuss the Trustworthiness of Digital Evidence
3. Discuss Detection and Recovery of Hidden Data
4. Discuss Digital Forensic Science in Networked Environments

More detail on these points will now be considered.

#### **3.4.1 New Digital Forensic Science Definition**

The members of the DFRWS argue that digital forensic science is defined as the use of scientifically derived and proven methods for the preservation, collection, validation,

identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources, specifically when it used for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorised actions shown to be disruptive to planned operations. This definition is in concordance with the definition used in this thesis for computer forensics (see introductory paragraphs of Chapter 2). Digital forensic is inclusive of computer forensics. Digital forensics was modelled as a linear process with steps. This is shown in the Figure 8 below. The grey boxes at the top of their matrix were identified as core processes, categories, or classes. The contents of the columns below each category are candidate techniques or methods belonging to that class.

Digital evidence was not inherently untrustworthy but integrity and fidelity of data were questioned. It was also agreed that human interaction with digital evidence was determined to be a fact of life in digital forensic science into the foreseeable future. Hidden data were classified in several general categories shown in figure below. The grey area shows some of the places where the data can be hidden. The workshop only classified the data and did not propose any models to identify the data being classified. As we can see the separation of core process with extraneous framing helps us to take this model to next level by systematically detecting AF activity at each stage of investigation.

The DFRWS model was an important milestone in the developing field of computer forensics and it significantly influenced the subsequent models and developments. The strength of the model is the acceptance it gives to human interaction and able optimise human involvement. The model specifically addresses ways in which data is hidden in forensic investigations. This will be

important when meta-forensic adaptations are considered later. Data hiding techniques are one of the four anti-forensic techniques this dissertation will try to account for.

Identification	Preservation	Collection	Examination	Analysis	Presentation	Decision
Event/Crime Detection	Case Management	Preservation	Preservation	Preservation	Documentation	
Resolve Signature	Imaging Technologies	Approved Methods	Traceability	Traceability	Expert Testimony	
Profile Detection	Chain of Custody	Approved Software	Validation Techniques	Statistical	Clarification	
Anomalous Detection	Time Synchron.	Approved Hardware	Filtering Techniques	Protocols	Mission Impact Statement	
Complaints		Legal Authority	Pattern Matching	Data Mining	Recommended Countermeasure	
System Monitoring		Lossless Compression	Hidden Data Discovery	Timeline	Statistical Interpretation	
Audit Analysis		Sampling	Hidden Data Extraction	Link		
Etc.		Data Reduction		Spacial		
		Recovery Techniques				

Grey	Core Processes
Yellow	Extraneous Framing

**Figure 8: Investigative Process for Digital Forensic Science (reproduced) [48]**

Graphics	Signals	Applications	Disk Geometry	File Systems	Comm Structures	Solid State	Data Structures	OS & Programming	Non-Digital
Least Significant Bit	Altered Compression Algorithms	Compound Doc formats	Marked Bad Clusters	Distributed Systems	Reserved Packet offsets	BIOS	Heap Space	Virus-like expressions	Perception
Audio	Stego	metadata - reserved structures	Maintenance Track	RAM Slack	Email Spam	CMOS		Rootkits altering system calls	File names
Video	Timing channels	File Slack	Extra tracks	Modified Dir Entries	Protocols	RAM		System Libraries	Plain sight
Imagery	sequencing		Hidden partitions	Unallocated Space				DLL's	
Stego				Boot Sector					

**Figure 9: Categories of Data Hiding (reproduced) [48]**

### 3.5 An Abstract Process Model

Reith, Carr and Gunsch [49] derived a model from the DFRWS model. There were nine basic components [49]: Identification, Preparation, Approach Strategy, Preservation, Collection, Examination, Analysis, Presentation, and Returning Evidence. The model attempted to standardise traditional forensic evidence collection strategy as practiced by law enforcement. The Preparation Phase should be before the Identification Phase so that the plan is in place when the incident is detected. The shortcoming of the model for the purposes of meta-forensics is that it does not define system level resources and ways it can be integrated into the model. This model uses many of the same phases as the DFRWS model described first, and only adds a description for each phase.

### **3.6 End-to-End Digital Investigation Process**

Stephenson [50] also used the DFRWS framework. Each process is classified as a “class” and actions taken as “elements” of the class. The model then states six classes defining the investigative process. The process is then extended to nine steps which are called the End-to-End Digital Investigation Process, or EDDI. The End-to-End process details consist of: Collecting evidence, analysis of individual events, preliminary correlation, event normalising, event deconfliction, second level correlation (consider both normalised and non-normalised events), timeline analysis, chain of evidence construction and corroboration (consider only non-normalised events). He then developed a formal representation of this process using Coloured Petri Net Modelling [51] for simulating the evidence. A new language called DIPL (Digital Investigation Process Language) was proposed to allow a structured description of the investigative process. DIPL language is like Lisp programming and was used to create formal model. DIPL was used in validating an investigation or investigative process but the language only worked for post-incident cause analysis. In chapter 5 we use one such method. This model was advancement as it permits formal verification unlike the preceding models. Any state changes that occurred during the course of the event were clearly represented without providing technical details of the incident. The meta-forensic model addresses this by providing technical details and advancement of DIPL language which looks at the properties of the events by weights which makes it a pre incident analysis tool or a threat detecting procedure as well.



### **3.7 Event-based Digital Forensic Framework**

A new approach was taken by Carrier and Spafford [52] who added several new elements to the digital forensic framework including event and events reconstruction. When a security breach is investigated, it is necessary to be able to reconstruct one or more events, this event reconstruction and hypothesis-testing phase was applied in Carrier and Spafford's model. Reconstruction also provides better understanding of events and helps to identify the dependencies, cause effect of a situation. A digital event was defined by them as: *“an occurrence that changes the state of one or more digital object. If the state of an object changes as a result of an event, then it is an effect of the event. Some types of objects have the ability to cause events and they are called causes.”*

The event-based framework was used to develop hypotheses and answer questions about an incident or crime. Hypotheses are developed by collecting objects that may have played a role in an event that was related to the incident. Once the objects are collected as evidence, the investigator develops hypotheses about previous events at the crime scene. This framework is based on the process model and is one of the most complete models proposed so far. The event-based framework particularly lends to meta-forensic investigation. Moreover, because meta-forensics is often concerned with testing whether attempts have actively been made to invalidate evidence, hypothesis generation and testing are vital to the model make up.

### **3.8 Case-Relevance Information Investigation Framework**

The Case-Relevance Information Investigation Framework is built upon the work of Carrier and Spafford [52]. Case-Relevance is defined in this framework as: “*the property of any piece of information, which is used to measure its ability to answer the investigative ‘who, what, where, when, why and how’ questions in a criminal investigation.*” An automatic and efficient framework was introduced [53] to provide the Case-Relevance information by binding computer intelligence technology to the current computer forensic framework. Knowledge reuse and sharing in computer forensics is also introduced. The framework used this notion to describe the distinctions between computer security and forensics defining degrees of case-relevance. This is shown discussed further in the next section. Knowledge reuse and automating aspects of case-relevance added investigatory power to this model and allowed the system to harness technology in an effective way in the forensic process. The human factor is still required as the DFRWS model originally maintained, and it seems unlikely that the process could ever become fully automated. But what the researchers did in this instance was to automate aspects in the relevant respects to augment the investigatory process. Due to the enormous complexity of data hiding, artifact wiping and other anti-forensic techniques, an efficient way of determining case-relevance proved crucial. But there were further improvements to come in next section.

### **3.9 Forensic Modelling Comparison and Outcome**

All the models of investigation follow a similar approach of Acquisition, Identification, Evaluation, and Admission as evidence within a framework. The model discussed shows the state of the current forensic investigation process. Although all the investigation follows a

common approach with each model the functionality and investigation process changes. Pollit model deals with identification in data collection phase while in DFRWS model it happens under incident response. This is represented in Table 1 below which compares and contrasts between various models. We can see from the table most of the investigative phases are common to earlier models and the approach to the forensic investigation process changes over time.

Model	Preparation	Data collection	Data Analysis	Incident Response	Finding present	Incident Closure	Integrity Check
<b>Approach to Evidence in Cyberspace (Pollit)</b>							
Acquisition							
Identification		Yes					
Evaluation			Yes				
Admission as evidence							
<b>Three-Level Hierarchical Model</b>							
Principles of examination		Yes	Yes				
Policies & practices			Yes		Yes		
Procedures & techniques	Yes	Yes	Yes		Yes		
<b>First Research Road Map 2001 (DFRWS model)</b>							
Identification				Yes			
Preservation							
Collection	Yes						
Examination	Yes						
Analysis			Yes				
Presentation			Yes		Yes		
Decision						Yes	
<b>Abstract Process Model Reith, Carr &amp; Gunsch</b>							
Identification				Yes			
Preparation				Yes			
Strategy for Approach				Yes			
Preservation		Yes					
Collection		Yes					
Examination			Yes				
Analysis			Yes				
Presentation					Yes		
Returning evidence						Yes	
<b>End-to-End Digital investigation Process</b>							

Collecting evidence	Yes	Yes					
Analysis	Yes	Yes					
Preliminary correlation			Yes				
Event normalising			Yes				
Event deconfliction				Yes	Yes		
Second level correlation				Yes	Yes		
<b>Event-based Digital Forensic Framework</b>							
Readiness Phases	Yes						
Deployment Phases		Yes					
Physical Crime Scene Investigation Phases		Yes	Yes				
Digital Crime Scene Investigation Phases				Yes	Yes		
Presentation Phase				Yes	Yes		
Hypothesis testing Phase			Yes	Yes	Yes		Yes
<b>Case-Relevance Information Investigation Framework</b>							
Absolutely Irrelevant	Yes				Yes		
Probably Irrelevant	Yes				Yes		
Possibly Irrelevant	Yes				Yes		
Possibly Case-Relevant	Yes				Yes		
Probably Case-Relevant	Yes				Yes		
Provably Case-Relevant	Yes				Yes		

**Table 1: Forensic models & the investigation steps**

When this model is applied to any forensic investigation some of the functionality is always lost as none of the models have all the required parameters to perform the investigation. The key element to any data gathering investigation is the validity of the gathered evidence. All the models failed to address this key issue except Event-based Digital Forensic Framework which had an integrity checker in hypothesis testing phase (Table: 1). This alone is not enough to validate forensic evidence as the rigorous AF activity detection is never considered in any of the steps or models.

Our focus is on AF and the clearly the current models and process have no procedure in place for detecting anti-forensics. The Basics of Forensic Investigation (Section 1.2) is questioned which

leads us to the hypothesis of risk of potential undetected anti-forensic activity in the system. The risk increases as the current model and procedures lack AF detection capabilities.

The forensic models alone cannot address the anti-forensic issues. For a successful AF exploit to happen the security of the system has to be compromised. So we review some of the security and reasoning models in our literature. The vulnerability in the security models also increases the chance of AF activity.

### **3.10 Policy Based Security Model**

A computer policy is set of acceptable rules set by an organisation for use of its computers and electronic infrastructure. A security policy is a statement that specifies what is allowed and what is disallowed with regards to security. Security policies partition the states of a system into a set of authorised or secure states and unauthorised or insecure states [54]. This can be represented by binary quantifiers of True or False. A well-known example of a security policy for a university system deals with confidentiality of classified data. The security goal for this type of system is that the system should prevent unauthorised disclosure or theft of the digital information. Enforcement is mandatory so that everyone dealing with classified data must follow these rules. The policy step is important in computer and network forensics. In anti-forensic activity the perpetrator would attempt to make it appear that the policy has not been broken, e.g. switching a True state to a False state. A meta-forensic investigator would be required to determine whether the detection system has been manipulated post-crime.

### **3.11 Bell-LaPadula Model**

The Bell-La Padula Model [55] was proposed to formalise the U.S. Department of Defence's multilevel security policy. The model is a formal state transition model of computer security policy that describes a set of access control rules that use security labels on objects and clearances for subjects. Security labels range from the most sensitive, Top Secret, down to the least sensitive, Unclassified or Public [55]. The restrictions imposed by this model is reading down and writing up.

### **3.12 Biba Model**

The Biba Model is a formal state transition system of computer security policy that describes a set of access control rules designed to ensure data integrity. Data and subjects are grouped into ordered levels of integrity. The model is designed such that subjects may not corrupt data in a level ranked higher than the subject, or be corrupted by data from a lower level than the subject [56]. This model was developed to remove the weakness in the Bell-LaPadula Model which only addresses data confidentiality.

### **3.13 Take-Grant Protection Model**

The Take-Grant protection model is used to establish or disprove the safety of a given computer system that follows specific rules. It shows that for specific systems the question of safety is decidable in linear time. The model represents a system as directed graph, where vertices are either subjects or objects. The edges between them are labelled and the label indicates the rights

that the source of the edge has over the destination. Two rights occur in every instance of the model: take and grant. They play a special role in the graph rewriting rules describing admissible changes of the graph. There are a total of four such rules: The take rule allows a subject to take the rights of another subject, while the grant rule allows a subject to grant its own rights to another subject. The create rule allows every subject to create new nodes, while the remove rule allows a subject to remove rights it has over another object. Using the rules of the take-grant protection model, one can reproduce in which states a system can change, with respect to the distribution of rights. Therefore one can show if rights are violated with respect to a given safety model [57].

### **3.14 Security Model Comparison and Outcome**

The above models all have their shortcomings and a successful exploit of one of it can lead to AF. Bell-LaPadula model lacked functionality to deal with integrity of data which restricted a subject from writing to a more trusted object. The Biba model addressed the problem but did not support the granting and revocation of authorizations. Another problem is that the model is used strictly for integrity and does not enforce data confidentiality. The Biba model cannot be used for authorization or revoking of privileges like in database. Hence the development of Take Grant model to overcome the limitations. The Take Grant model has its limitations. Take Grant model did not consider the issue of integrity. The model is limited to number of nodes that can be shown at one time as it used directed graphs for representation.

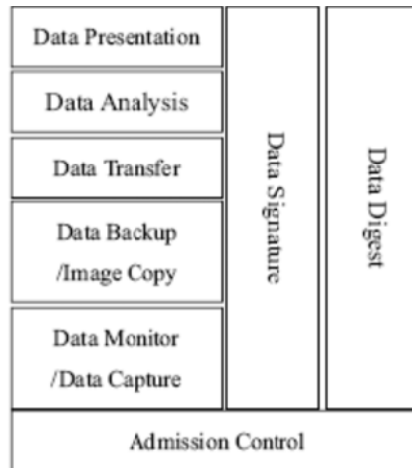
None of the models addressed security completely. Combinations of models were used to implement security. This approach led to more complexity when a system is compromised and the vital AF activity goes undetected. Dorothy Denning [58] pointed out that models have theoretical limits and cannot always prove that a model satisfies certain security conditions. The models are based on strict mathematical properties can lead to systems that are vulnerable. In the following Chapter we show that this threat can be exploited. This research tries to address this issues using meta-forensic model where the security design issues are considered during assigning of weights in Chapter 5 to detect AF activity. There are number of other security models in existence which is not discussed in literature, but the issues identified above equally applied to all of them.

### **3.15 Modelling Network Forensic Processes**

The forensic and security model was an important development in computer forensics and security with the ever changing environment, it came just at the right time as a new version of digital crime was quickly being identified – network attacks. Network forensics uses forensic models to solve network security issues. Wei Ren and Hai Jin [59, 60] discussed the network forensics model and its fundamental fields, such as taxonomy, conceptual model, legal principles, key techniques, canonical processes, forensics systems architecture and deployment. Standardisations of network forensic processes were also proposed and a prototype was implemented by Wei and Hai. The general process of network forensics includes five steps. They are ‘capture, copy, transfer, analysis, and presentation’. Their prototype was implemented using open source tools. The most important function in this system is data analysis. Figure 10 below



shows the general computer network forensics system architecture is given. Once the complete data is captured, the analysis machine system begins the analysis.



**Figure 10: General Computer Network Forensics System Architecture**

The model is significant that it deals with the more complicated problems of identifying and locating network security issues in this respect. The difference between computer and network forensics is multiple layers of detection capability compared to computer forensics. As in computer forensics most of the breach identified is post incident compared to network forensics where active AF detection is possible. This however does not rule out advanced antivirus software's detecting security breaches in computer but to identify and analyse AF activity network forensics presents more information for better decision making. The current network forensic model was only intended to be a high level model and the working prototype was not validated by experimental data. The model did not define data structures. This was overcome in Common Information model (CIM) [61] which attempted to define specification and schema for system integration. Advancements needed to be made in addressing the integrity of collected

data and meta-forensic model tried to address this issue. The network forensic model provided an input into meta-forensic model system automation discussed in Chapter 5.

### **3.16 Vulnerability Tree Model**

The way vulnerability of any system is addressed plays a key role is identifying threats in a system. This threat when materializes may result in AF when not detected by standard approaches. Vulnerability Tree Model proposed by Vidalis and Jones [62] to evaluate threats in a computer based system. They describe vulnerability trees as *“hierarchy trees constructed as a result of the relationship between one vulnerability and other vulnerabilities and/or steps that a threat agent has to carry out in order to reach the top of the tree”*. The top of the tree is known as the top vulnerability or the parent vulnerability and is symbolised as capital ‘V’. Each of these constitutes a branch of the tree. The branches are constructed by child vulnerabilities. Consequently the child vulnerabilities can be exploited by steps that the threat agent will have to perform in order to get to the parent. The child vulnerabilities are denoted by the lower case ‘v’ and the steps with the lowercase ‘s’. Each vulnerability is broken down in a similar way. Normally this will end up in more than one level of decomposition. When the point is reached where the branches contain only steps, and no child vulnerabilities, then we know that we have reached the lowest level of decomposition. It’s called the “step-only” level. The Vulnerability Model contributes a dimension absent in the network forensic model, where on a multi-tiered level one is able to assess the impact of agents exploiting aspects of the system. This vulnerability architecture is helpful when we traverse down the AF activity tree to identify the AF threat. This can be advancement to the vulnerability tree model to detect AF.

### 3.17 Reasoning and Logic

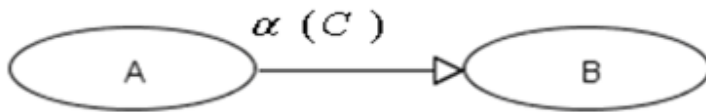
With model and vulnerability taken into account for AF at this stage of the literature review, a step forward is taken by reasoning the cause, effects and circumstances surrounding it on the analysed system. The reasoning implemented behind the models forms the backbone to assign weights to a given set of conditions. It is important to consider this because assumptions made behind the logic of the models ultimately will form the effectiveness of the model. We take different logical approach to determine if we can arrive at the same conclusion during assignment of weights discussed in Chapter 5.

#### 3.17.1 Extended Finite State Machine

**Approach 1:** The analysed system is considered as an Extended Finite State Machine (EFSM) model, which can be expressed by an ‘if statement’ consisting of a set of trigger conditions. If trigger conditions are all satisfied, the transition is fired, bringing the machine from the current state to the next state and performing the specified data operations” [63]. All possible scenarios of the incident can be determined by backtracking transitions leading to that state. We start by Obtain a finite state machine under investigation. Determine all possible scenarios of the incident by backtracking transitions. Discard scenarios that disagree with the available evidence.

To compare and contrast an event validation requires a Boolean condition to determine if the system state has been changed from the time a FSM is obtained for investigation. So we use extended finite state machine (EFSM) modelling in which the transition can be expressed by an

“if statement” consisting of a set of trigger conditions. If trigger conditions are all satisfied, the transition is fired, bringing the machine from the current state to the next state and performing the specified data operations. This can be represented by a state transition diagram shown in Figure 11. The initial figure represents the system “when event  $\alpha$  occurs in state A, if condition C is true at the time, the system transfers to state B”.

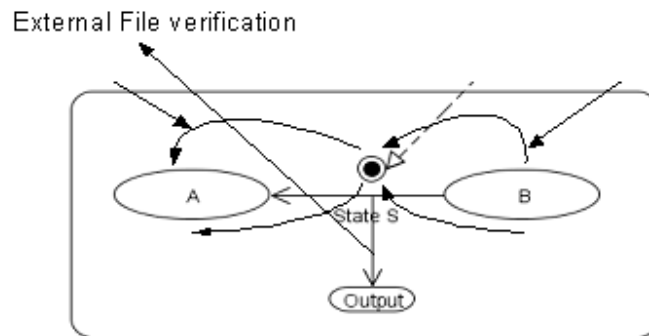


**Figure 11: Finite State Transition**

The state A is assumed to be the normal working of a system and state B is the compromised system under investigation. The condition C is assumed to be the compromise or "hack" which alters the system state. Condition C leads to system state B which is determined by the system investigator as the system under investigation.

Pavel Gladyshev's [64] model followed a backtracking algorithm to reconstruct the events to determine why it happened or who may have done it. This reconstruction approach in our research forms a component of meta-forensic model. The Boolean condition triggered during the event links the evidence to all the available external and internal sources. During the event retract a history mechanism checks for the last state and verifies if it was linked to any other files (internal or external) when the event happened and the change is reported as rewritten. The output is shown in Figure 12. The external verification file can be a backup file which was taken during normal system operation. The chart represents the last visited state where the Boolean

operation is performed. The outcome of the Boolean operation determines if the evidence was compromised.



**Figure 12: EFSM representing Boolean conditions for verifying change**

Pavel Gladyshev's reconstruction with backtracking was incomplete as the knowledge of the system functionality was not considered. This is overcome in meta-forensic model.

### 3.17.2 Counterfactual Reasoning

**Approach 2:** Counterfactual reasoning [65, 66] is used after the events are identified. It can be used on MES-diagram shown in Figure 13. It is applied on every pair of events in the diagram and an arrow can be drawn to show their dependency. System compromise is the cause of the investigation followed by X, Y and Z where all the objects, similar to the first, are followed by objects similar to the second. If X has not existed Y and Z would have never taken place. MES-diagrams have two axes. The horizontal axis represents time. The Vertical axis lists actors involved in the accident. Event blocks are placed on the diagram according to their time and actor. The interrelationship between events shows the forensic model being followed with the

timeline. What this approach lacked was the functionality of Boolean operation being applied to their dependencies. We consider in next approach.

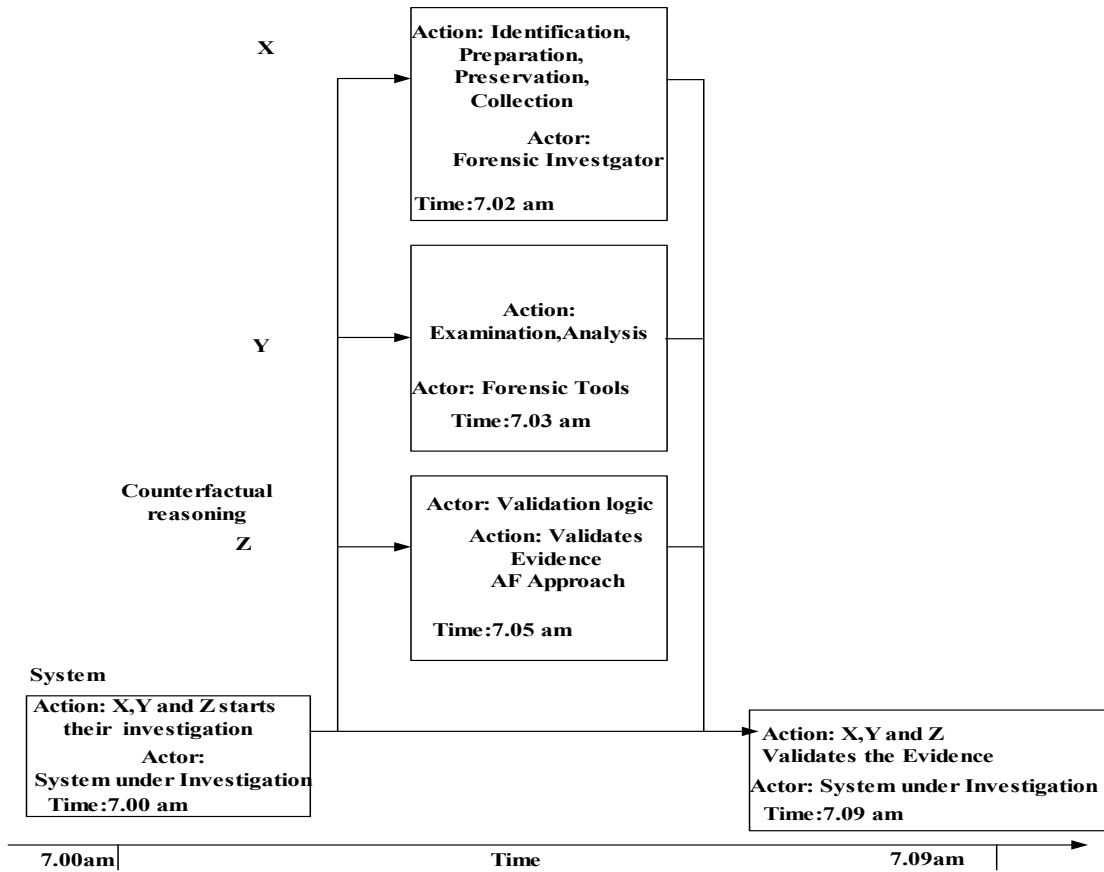
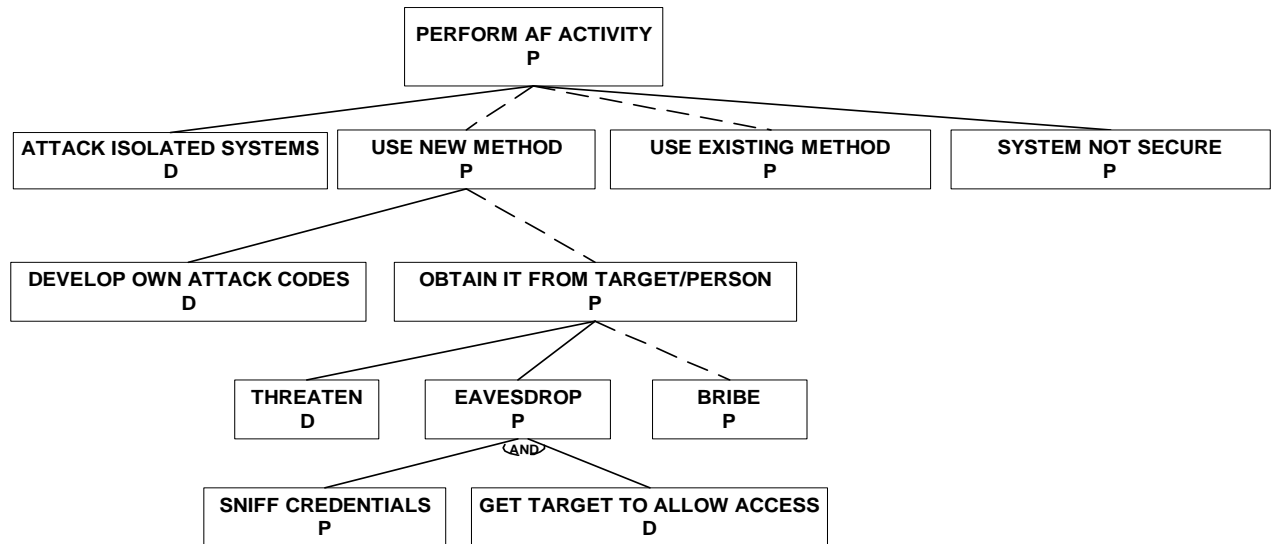


Figure 13: MES Diagram for validating data

### 3.17.3 Attack trees and Vulnerability tree Reasoning

**Approach 3:** The file validation mentioned in above section is possible when the system being investigated is defined. An important part of validation process is identification of possible

incident scenarios and threats. The threats are identified using the vulnerability tree model discussed in previous section and scenarios by attack tree. Attack trees are conceptual diagrams of threats on investigated system and possible incident scenarios [67].



D-DIFFICULT

P-POSSIBLE

**Figure 14: Attack tree describing different ways to Perform AF activity**

Solid Lines	AND Operations
Dotted Lines	OR Operations

The aim of this work is represented by the root node. Other nodes represent sub goals that must be achieved to achieve the main goal. The figure above shows an attack tree of trying to perform an AF activity without authorisation. The goal of the tree is to compromise a system and successfully perform AF activity. The goal can be achieved in a number of different ways, such as use new methods, use existing methods or combination of both. The basic attack tree is built from two types of node: AND nodes, and OR nodes. In the figure, everything that isn't an AND node is an OR node. To meet OR node conditions, any one of its child nodes must be fulfilled.

To meet an AND node, all of its child nodes must be fulfilled. The nodes can take Boolean value, probability of possible or difficult or impossible. The analyst must use intuition and common sense to build up a generic tree structure. Similar logic is followed in forensic investigation but is not defined in a structured way and this thesis tries to define it as meta-forensic model.

### 3.17.4 Why-because analysis

#### **Approach 4:**

Why-Because Analysis (WBA) [68] is a rigorous technique for causally analysing systems. It is used for the analysis of accidents, mainly in transportation systems (air, rail, and sea). This concept is required for the validation approach because it represents a set of concepts within a domain and the relationships between those concepts. It is used to reason about the properties of the domain in AF, and can be used to define the domain. The detection of AF activity can be represented as cause effect graph [69] which is a directed graph that maps a set of causes to a set of effects. The causes may be thought of as the input to the program like saving a file in a system and the effects may be thought of as the output, like increased storage. The cause-effect graph shows the nodes representing the causes on the left side and the nodes representing the effects on the right side. There may be intermediate nodes in between that combine inputs using logical operators such as AND and OR.

Causes-----> Nodes (AND, OR) -----> Effects

The validation of any evidence requires an input and can be defined using this concept to generate a decision table.

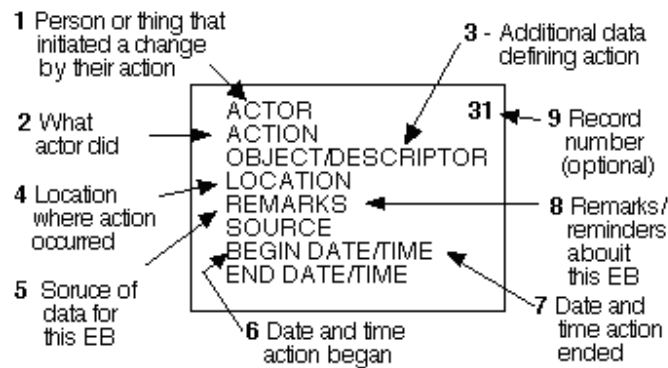


### **3.18 Multi Linear Events Sequencing (MES)**

The Multi Linear Events Sequencing model, proposed by Benner [70], is similar in many respects to an attack tree model and reasoning techniques discussed in previous section. This model is combination of model and logic .It is used in conducting accident investigation. Benner advocates close attention to the sequence of events leading up to the accident, with special status given to the temporal relations between events. This principle is similar to the AF theory of validation to capture the events leading up to the system compromise. The basic component of the MES process is the data structure on which all other elements rest. This data structure is defined by the Event Building Block (EB). An Event Building Block is defined as one action by one person or object.

1 actor + 1 action = 1 Event Building Block (EB)

A simple actor/action EB works fine for simple investigation. But for a forensic investigation it is necessary to use more data elements during complicated occurrences involving more than two or three actors. Figure 15 below shows the additional elements which are necessary for forensic investigation.



**Figure 15: Additional elements for MES**

The EB and additional elements are used to list events in Chapter 5 where meta-forensic model is proposed.

### 3.19 Summary

The above sections discussed literature review as three stages. It discussed forensic models followed by security models to represent the current state of affairs. The forensic and anti forensics have been looked at from the model prospective. In final sections the logic behind reasoning of a given condition is discussed.

Stages of forensic investigation discussed in section 3.1 to 3.9 shows the evolution of the process and the framework. The reviewed models dating back from 1995 (Pollit) up till now shows the development of forensics. Section 3.9 discusses the comparison of forensic models. The literature identifies a gap in knowledge where none of the models address AF issues or takes into consideration the concept of validation.

From Section 3.10 onwards Security models and their shortcomings are discussed. One of the key issues identified was the threat this poses to the current system which may result in AF activity. The Biba Model, Bell-LaPadula Model, Network forensic models all have their own limitation which brings us back to the hypothesis of this thesis "There is always a risk of potential undetected anti-forensic activity in the system" and this risk increases with vulnerable models. The literature has identified this issue and we appraise this by proposing solution later in this thesis. The appraisal of the current model is only possible by proposing a new meta-forensic model which works alongside with all existing models.

Final section of literature reviews the logical reasoning of how an AF scenario can be analysed. Each approach adds a functionality which is helpful later on to assign weights .The reasoning logic plays out the events which helps us identify and define threats. The literature review of logical reasoning shows that the problems of real world incidents are better defined by reasoning. The gap in knowledge of reasoning approach is the limitations of using it mathematically .We appraise this logical reasoning by assigning weights to events so the mathematical evaluation of a given AF situation is possible.

## **Chapter 4: THE ANTI -FORENSIC EXPERIMENT**

The literature review attempted to establish the role of models in computer forensics and security. The discussion focused on the requirements needed to address anti-forensic attacks. There is a growing concern about data integrity given anti-forensic activity and there appears to be a need for a meta-forensic process to exclusively deal with anti-forensic activity over and above the extant procedures for forensics and security. A meta-forensic step would further the advances already made in securing data integrity. Meta-forensics is seen to form the next step on a continuum, as opposed to a discrete step in its own right. The aim of this chapter is to establish the problem statement that will inform the experiments in Chapter 5. The problem is that the integrity of digital forensic data is compromised by the mere threat of anti-forensic activity. It doesn't matter if anti-forensic activity is detected or not. The threat is enough and in this Chapter we show various ways the threat is exploited and develops into an AF case. Here we are not detecting AF but showing the AF as it happens in a given system. The analysed literature in previous section exposes the gap in knowledge of how the systems can be insecure and in this Chapter we demonstrate by way of experiment that the identified shortcoming through practical experiments.

### **4.1 Components of the Experiment demonstrating anti-forensic data compromise**

For most research projects the choice of research methods is heavily influenced by methods other Researchers in the field have chosen. For anti-forensics field, other researchers have presented analysis and classification in form of studies. These case studies only classify it (Section 2.5)

without any series of controlled experiments. In this thesis we demonstrate the AF activity. We focus our attention particularly before or during the evidence is collected.

**Aim:** In the following experiments it will be demonstrated that anti-forensic is possible for a given system even when the forensic investigation model procedures are followed.

The experimental setup was creation of VMware infrastructure consists of a dual core AMD Turion box with 4GB RAM, 250GB Hard disk and Microsoft Windows XP Service pack 3 as operating system. VMware server version 2.0 was installed. Four virtual machines with Linux, Windows 2003, Windows 2000 and Windows XP were installed. Each node was deployed with 20GB hard disk space and 2 GB RAM. More about VMware is detailed in Appendix B. Additional tools and commands used in the experiment are defined in subsections.

This set-up, though not comparable to large production network, resembles similar networks and was sufficient to provide a test bed for the reviewed security software tools. Prior to running each experiment the environment is “reset”. For the virtual environment, a VM snapshot was taken, and for the native environment, a mirror of the native OS was taken. This approach allows for easy restore after each experiment.

## **4.2 Testing framework**

The framework used in this experiment follows two test procedures and methodologies for demonstrating AF.

#### **4.2.1 NIST CFTT**

The aim of the Computer Forensic Tool Testing (CFTT) project at the National Institute of Standards and Technology (NIST) [71] is to establish a methodology for testing computer forensic software tools by the development of general tool specifications, test procedures, test criteria, test sets, and test hardware. CFTT is one of the most comprehensive forensic tool test frameworks available. Since there are not many frameworks available the current experiment will contribute towards a new testing framework based on CFTT standards.

CFTT has divided its activities of forensic investigations into categories, and then developed a test methodology for each category. The test methodologies follow a standard validation and verification procedure. The test is divided into generic requirements followed by test cases and then a tool-specific test procedure.

#### **4.2.2 Digital Forensic Tool Testing (DFTT)**

The DFTT project that was initiated by Brian Carrier in 2003 and was aimed at bridging the gap between the comprehensive test developed by the CFTT and the needs of practitioners [72]. The projects, named Digital Forensics Tool Testing, collect test cases contributed by practitioners in the field. There are currently fourteen test cases available (the latest addition was in August 2010). Using similar test cases provides a measure of assurance that the tools used in the investigations of computer-related crimes produce valid result.

### **4.2.3 Combination method**

This is the test method devised by using the combination of NIST and DFTT method. We call this the combination method as it combines all known test processes. Using the combinational method we not only test tools but also test the methodology and framework discussed in literature review. This method enhances the capability of both methods. The steps followed are:

1. Acquire all required tools to be tested.
2. Review tool documentation.
3. Install tools in different versions of operating systems like Linux RHEL5, Windows.
4. Select relevant test cases depending on tool functionality.
5. Decide the network and server layout.
6. Execute tests.
7. Test results.

The test method is intended for rigour, despite perhaps seeming very obvious. Thus, this will be the framework used in this experiment.

### **4.3 Combination method implementation**

The security tools can be classified in number of ways. For this research we tried to classify them into the following categories. The tools of interest for this research are also listed.

- 1) Forensic tools –Encase, TCT, The Sleuth Kit
- 2) Anti-forensic tools – Srm, Evidence Blaster, Metasploit
- 3) Vulnerability detection tools (Operating system\Network) –Retina, NMAP

- 4) Vulnerability prevention tools (Operating system\Network)
- 5) Vulnerability exploitation tools (Operating system\Network)
- 6) Packet sniffing tools
- 7) Packet crafting tools
- 8) Intrusion detection system tools (Operating system\Network) - Snort
- 9) Password cracking tools
- 10) Disassembler tools
- 11) Traffic monitoring tools
- 12) Data recovery tools - TestDisk Utility, e2undel

The classification does not limit the tools to be in other categories as all the security tools can be used for forensic and anti forensic purposes. We use these tools to experimentally review the functionality of models discussed in literature and demonstrate AF activity.

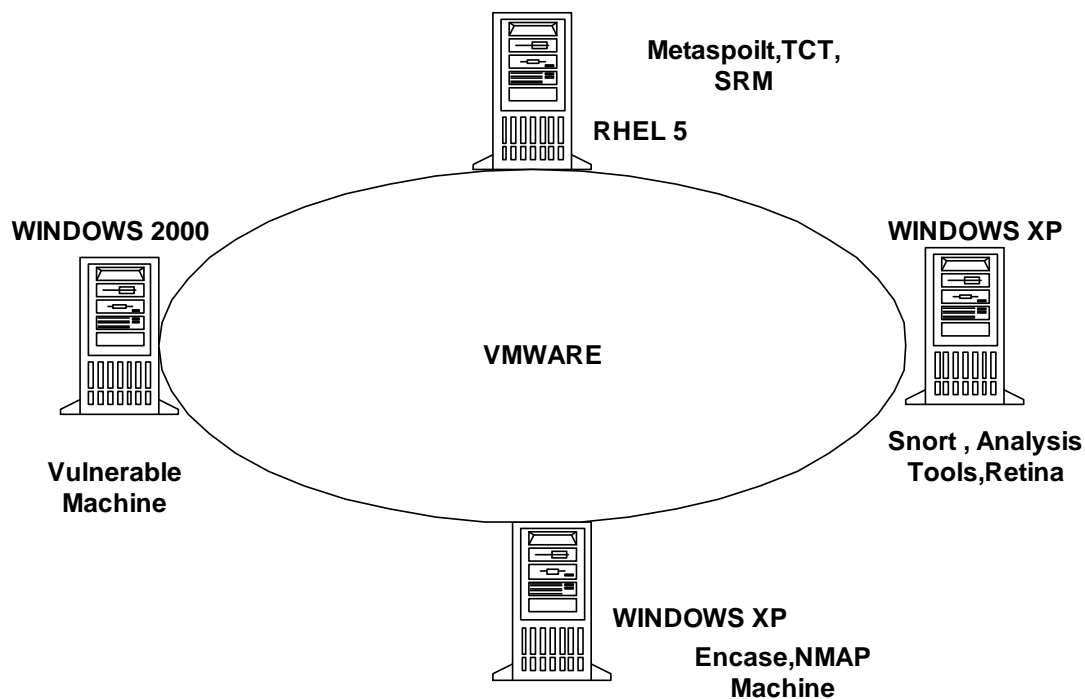
#### **4.4 The Experimental Setup**

STEP1: In our experimental setup we use token ring topology network discussed in section 2.3.1 with Win 2000, RHEL 5 and Windows XP SP1 OS. The experimental setup is shown in Figure 16.

STEP2: The Metasploit Framework, Srm, e2undel, TCT, The Sleuth Kit was installed in RHEL system.

STEP3: Snort, all the log analysing software, Encase, Vulnerability scanning software Retina was installed in Win XP system.





**Figure 16: Experimental setup for anti-forensic activity**

With the test bed setup detailed we now start our experiments .The tools in the test bed were only installed as and when necessary for the performed experiment.

#### **4.4.1 Secure remove (SRM) and TestDisk Utility**

Srm (secure rm) is a command-line tool rm which destroys file contents before unlinking the inodes in the file. This prevents the users from recovering deleted information when the machine is compromised. This tool was installed in Linux as detailed in Step 2. A test case “test. file” was used to perform this experiment. To create this block file the test file was downloaded from the test case mentioned in section 4.2.The install command script(./install-sh) was used to install the

file and `./test.sh` command was used to create test block files. Once the test.block files were installed and created the `srm delete` command was used. The process is shown in Figure 17. This prepares our test environment.

From our model assumption discussed in literature (section 3.7) of Event-based Digital Forensic Framework we should be able to recover the deleted files. There are two assumptions made at this stage

- a) Assuming that the secure delete has happened.
- b) The deleted files are recoverable.

We run a list command `ls -lrf` to verify the file `test.block` is deleted. We conclude that the file is deleted and is not listed under hidden file. To recover the file test disk utility is run. Testdisk is powerful data recovery software. It was designed to help recover lost partitions and we use it to try and undelete files. The tool was not able to recover the deleted block. The tool not being able to recover the file doesn't prove that the file is unrecoverable – all it shows is that this particular tool was unable to recover the deleted block. A different tool `e2undel` was employed to recover the file however the tool failed to list or recover the deleted file.

So the assumption that the deleted files are recoverable is incorrect and in an investigation not all deleted files are recoverable. The model assumptions from literature following the investigation procedure will assume the forensic investigator will have no knowledge of secure delete happening in system. So this brings us to the question of evidence discussed in Section 2.1.1 whether the evidence file exist or it does not when legality of evidence is considered. But we know by experiment that the file was deleted and there is every possibility of undetected AF.

```

root@dhcpc0:/downloads/srm/srm-1.2.10 (on dhcpc0)
File Edit View Terminal Tabs Help
config.log doc Makefile.in srm.spec test.symlink
[root@dhcpc0 srm-1.2.10]# ls
aclocal.m4 config.status INSTALL missing srm.spec.in TODO
AUTHORS configure install-sh NEWS srm.vcproj
ChangeLog configure.in lib README stamp-h1
config.h COPYING Makefile src test.file
config.h.in depcomp Makefile.am srm.sln test.sh
config.log doc Makefile.in srm.spec test.symlink
[root@dhcpc0 srm-1.2.10]# ./install-sh

[root@dhcpc0 srm-1.2.10]# ./install-sh /downloads/srm/

[root@dhcpc0 srm-1.2.10]# ./test.sh
creating symbolic link `test.symlink' to `test.file'
removing test.symlink
removing test.file2
removing test.dir/link2
removing test.dir/file2
removing test.dir
removing test.dir2/test.file
removing test.dir2
removing test.char
[root@dhcpc0 srm-1.2.10]# █

```

```

brw-r--r-- 1 root root 1, 1 Feb 27 10:09 test.block
[root@localhost srm-1.2.10]# srm -r test.block
[root@localhost srm-1.2.10]#

```

```

root@dhcpc0:/downloads/srm/srm-1.2.10
File Edit View Terminal Tabs Help
TestDisk 6.11.3, Data Recovery Utility, May 2009
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org
* Linux 0 1 1 12 254 63 208782 [/boot]
Directory /
drwxr-xr-x 0 0 1024 10-Feb-2010 11:33 .
drwxr-xr-x 0 0 1024 10-Feb-2010 11:33 ..
drwx----- 0 0 12288 10-Feb-2010 11:18 lost+found
drwxr-xr-x 0 0 1024 10-Feb-2010 11:57 grub
-rw----- 0 0 3297419 10-Feb-2010 11:33 initrd-2.6.18-164.el5.img
-rw-r--r-- 0 0 158 18-Aug-2009 19:56 vmlinuz-2.6.18-164.el5.hmac
-rw-r--r-- 0 0 954947 18-Aug-2009 19:56 System.map-2.6.18-164.el5
-rw-r--r-- 0 0 68663 18-Aug-2009 19:56 config-2.6.18-164.el5
-rw-r--r-- 0 0 107405 18-Aug-2009 19:56 symvers-2.6.18-164.el5.gz
-rw-r--r-- 0 0 1855956 18-Aug-2009 19:56 vmlinuz-2.6.18-164.el5

Use Right arrow to change directory, c to copy,
h to hide deleted files, q to quit

```

Figure 17: Secure Remove process

Gödel's incompleteness theorem (Section 2.1.1) for any evidence statement E= Files deleted in system, which states "there is no proof of E i.e. files deleted ". If E is true, there is no evidence. If E is false, there is a proof that evidence exist is true, which is a contradiction. So the hypothesis discussed in section 1.1 is a valid concern.

#### **4.4.2 Encase Evidence Verification**

For this experiment we use our experimental lab setup discussed in section 4.4 this experiment is carried out in XP system. From literature review, section 3.3 we know most of the forensic investigation follows models procedures and techniques .We show by way of this experiment that the procedure followed can have their own shortcomings. We use various tools for investigation and one such tool used for investigation is Encase which is designed to record forensic data stored on desktop PCs and servers and to recover deleted data. Encase is a proprietary forensic software produced by Guidance Software. Encase 6.1 was installed on a Windows machine. A test case was created by making a disk image of the system (a binary image F :). The data was acquired to case successfully.

```
Acquire
Status: Completed
Start: 08/25/11 01:09:21PM
Stop: 08/25/11 01:17:42PM
Time: 0:08:21
Name: F
Path: D:\F.E01
GUID: 82EE5A74BAD34846B639C6E451F724FB
Acquisition Hash: B3A0366A68F65E0B64E2C0B4EDF110CC
```

The test case image has 6 image file in a folder. The folder types along with their hash are listed below. To generate the hash we use Md5deep tool.

Md5deep is a set of programs to compute MD5, SHA-1, SHA-256. Md5deep was used as it can handle recursive operation by examining an entire directory tree. Md5deep can accept known hashes and compare them to a set of input files. The hashes for the five images are computed as shown in Table 2.

NO	File Name	MD5 hash
1	File1.jpg	e5faee69b323608aef127f6d9c933fc1
2	File2.jpg	0d4935134785b557852acfe2e924699d
3	File3.jpg	963cd543c7954f9f2059dfef2dbffeff
4	File4.jpg	959aa1dda097b2429bf9224c6babfaf3
5	Fileremove.jpg	37c94541c2175e6465bcdbeeffde1d7d
6	Filehide.jpg	83a2a71fdb52a46e1f3a106a03a9eece

Table 2: Files for Experiment

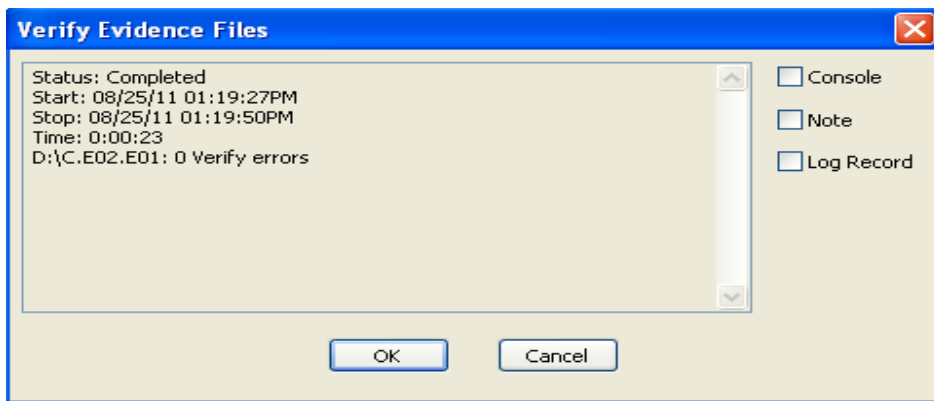
As the case file contains all the data any tampering after the collection of evidence (case F) can be detected using the function. So the CRC check only runs on case file not the original files.

The evidence file was validated using file check sum function in encase. This is shown in Figure 18 below.

In our experiment the original disk is subjected to tampering (Not the case file but Original F: disk). Fileremove.jpg was deleted using a Bdel tool which removes files and the associated tool as well. Information was hidden in Filehide.jpg

```
D:\EnCase6\MD5deep\md5deep-3.9.2\TEST >BDel.exe File1.gif.jpg  
D:\EnCase6\MD5deep\md5deep-3.9.2\TEST >BDel.exe BDel.exe
```

```
D:\EnCase6\MD5deep\md5deep-3.9.2\TEST>copy /B Filehide.jpg+HIDE.rar Filehide.jpg  
Filehide.jpg  
HIDE.rar 1 file(s) copied.
```



**Figure 18: Evidence Verification**

A new test case was created (Encase file:F1) using same parameters. The data was acquired to this new case file. The evidence file was validated again using file check sum function in encase but no error was detected. This is because the CRC test files only checks for tampering of binary evidence file (case F1).

Acquire

Status: Completed

Start: 08/25/11 01:23:50PM  
Stop: 08/25/11 01:32:13PM  
Time: 0:08:23  
Name: F1  
Path: D:\F1.E01  
GUID: 87099D08FB230E4F9351730C29E1BBD1  
Acquisition Hash: C521C3A38528EA27D38672407BCBAE26

Verify Evidence Files  
Status: Completed  
Start: 08/25/11 01:38:02PM  
Stop: 08/25/11 01:38:22PM  
Time: 0:00:20  
D:\C.E01: 0 Verify errors

The first time when we load an evidence file, EnCase will attempt to verify the data added to the case. When the data is captured, the checksum information is saved directly to the EnCase evidence file. This integrity verification process calculates the checksums in the evidence file and flags any data that has been altered. The check only happens in case level.

However inspecting Case F and Case F1 for evidence tampering using the calculated checksum we detected files being changed from previous run.

Filehide.jpg (35def5e25c80d4b1d98e601bfeb16c76) --- New Altered File

Filehide.jpg (83a2a71fdb52a46e1f3a106a03a9eece) --Old File

One can question what if the hash value was edited? If that was the case a successful undetected AF had taken place and the hypothesis in section 1.1 is a valid concern. Inspecting and searching for deleted file in hard disk, we were not able to detect deleted files. Inspecting the hard disk showed change in the disk space which suggest that a change has happened.

The conclusion of the above experiment is that the current model and procedure fails when same evidence is subjected multiple investigations. The tools used for investigation only detect AF at certain levels in investigation framework. As discussed in section 2.2, third party should be able to examine the evidence again if required and achieve the same result at any time. But the above experiment demonstrates that it's possible to obtain different results during investigation. This highlights a need for a validation method which can identify compromised evidence during regular evidence gathering.

#### **4.4.3 The Coroners Toolkit**

In this experiment we demonstrate one more data deletion from hard disk. The deleted data is attempted to be recovered by using the Coroners Toolkit (TCT) which is a collection of programs by Dan armer and Wietse Venema [73] for analysis of a UNIX system after system compromise. TCT captures access patterns of files, and recovers deleted files including cryptographic keys from process or from files.

The test case was a list of files and folder under the download directory called "casetct". The file permission was changed using standard Linux commands (chmod). One of the commands in the TCT tool was executed.

```
./grave-robber -d -V -c -E /downloads/casetct/-o RHEL
```

All the information about the running system was saved to a file. From the "casetct" directory few of the jpg images were deleted using *srm* command which securely removes files. The system was restarted to clear all the memory and running process. The tools in TCT suite



“*unrm*” and “*lazarus*” was run to try and recover the deleted file. The grave robber tool was run again to collect the information about the running system.

```
./grave-robber -d -V -c -E /downloads/casetct/-o RHEL-after-file-Del
```

The result: The recovery commands from the tool kit failed to recover the deleted files. The comparison of file output of *grave-robber* utility before and after file deletion shows the system state and missing files, which demonstrates that files can be deleted securely and still can remain undetected during forensic investigation.

#### **4.4.4 The Network AF**

In earlier experiments we demonstrated AF on individual system and in this section we expand this to network system. The experiment starts with scanning the network using NMAP and retina over a target IP range looking for known vulnerabilities, giving a potential attacker a quick idea of what attacks might be worth conducting. We used Retina Network Security Scanner written by eEye [74], as it contains all the integrated security and vulnerability management tools needed to effectively identify network vulnerabilities. The drawbacks of false positive and false negative rates are not considered in this research.

The scan identified vulnerability in each box in the network. A short overview result is shown in Table 3. Detailed results are included in Appendix C. Windows 2000 machine recorded 273 vulnerabilities. This does not imply that there is no vulnerable machine in network but the tools

employed didn't detect it or the vulnerability is not classified at time of this research. An attack packet was constructed using metamorph framework and deployed to both XP and Windows 2000 systems exploiting netapi vulnerability which is shown in Figure 19. The exploit provided remote code execution rights in Windows 2000 and in Windows XP.

Report Summary			
Scanner Name	Retina	Machines Scanned	1
▫			
Scanner Version	5.11.3.2195	Vulnerabilities Total	273
▫			
Scan Start Date	06/03/2010	High Risk Vulnerabilities	152
▫			
Scan Start Time	23:52:55	Medium Risk Vulnerabilities	64
▫			
Scan Duration	0h 1m 52s	Low Risk Vulnerabilities	57
▫			
Scan Name	FULL SCAN	Information Only Audits	26
▫			
Scan Status	Completed	Credential Used	6B3459551CAD439098C4B8AD6977FE10
▫			
Vulnerable Machines	1		

**Windows 2000 scan report**

### Report Summary

Scanner Name	Retina	Machines Scanned	1
▫			
Scanner Version	5.11.3.2195	Vulnerabilities Total	2
▫			
Scan Start Date	07/03/2010	High Risk Vulnerabilities	0
▫			
Scan Start Time	00:41:33	Medium Risk Vulnerabilities	1
▫			
Scan Duration	0h 12m 25s	Low Risk Vulnerabilities	1
▫			
Scan Name	FULL SCAN	Information Only Audits	1
▫			
Scan Status	Completed	Credential Used	289C25B453FC41B7A4A2A68E778201AB
▫			
Vulnerable Machines	1		

### Linux scan report

### Report Summary

Scanner Name	Retina	Machines Scanned	1
▫			

Scanner Version	5.11.3.2195	Vulnerabilities Total	0
□			
Scan Start Date	07/03/2010	High Risk Vulnerabilities	0
□			
Scan Start Time	01:10:54	Medium Risk Vulnerabilities	0
□			
Scan Duration	0h 5m 14s	Low Risk Vulnerabilities	0
□			
Scan Name	FULL SCAN	Information Only Audits	6
□			
Scan Status	Completed	Credential Used	CCFBD62133E24AFF98183FA10417F764
□			
Vulnerable Machines	1		

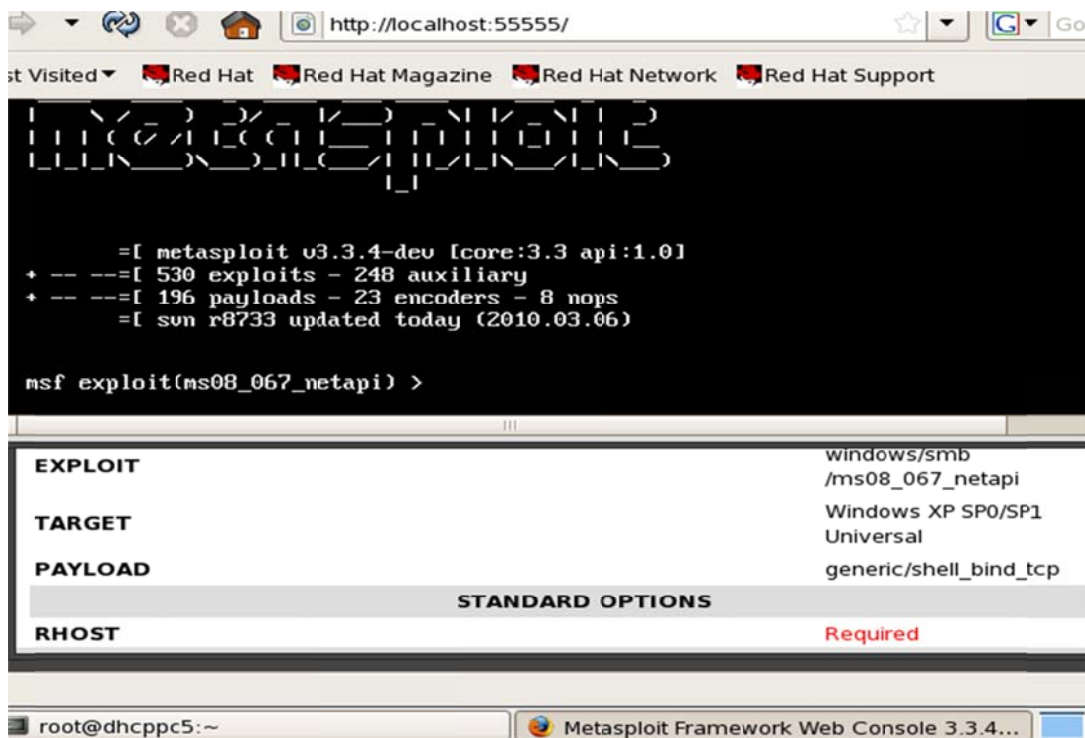
### Windows XP scan report

**Table 3: System Scan Reports**

Using the remote execution rights the event logs were deleted in target system. To detect the attack we used snort which is a network intrusion detection system (NIDS) and intrusion prevention system (NIPS) capable of performing packet logging and real-time traffic analysis on IP networks [75]. Snort is an able tool to analyse protocols, search or match files and actively block or detect a variety of attacks, such as buffer overflows, stealth port scans, web application attacks, OS fingerprinting attempts, Virus Trojans and worms attack on network and can also be used for intrusion prevention purposes, by dropping attacks as they are taking place. We

analysed the snort logs using software such as suguil, snortlog, snortsnarf to provide a visual representation of intrusion data.

Snort picked up the intrusion and flagged up an alert but failed to detect event logs deletion. Shown in Figure 19 is the result of the execution. As the XP system was not vulnerable the attack did not succeed however it left a trace of the attack in snort logs. The result can be analysed using MES diagram discussed in section 3.1. The experiment highlights the concept of security models described in literature and shows how a breakdown of one of the components can lead to successful AF.



I	Date	Time	Priority	Hostname	Message
	03-07-2010	02:57:39	Auth.Alert	127.0.0.1	snort: [1:402:8] ICMP Destination Unreachable Port Unreachable [Classification: Misc activity] [Priority: 3]; (ICMP) 192.168.1.7 -> 192.168.1.1
	03-07-2010	02:57:35	Auth.Alert	127.0.0.1	snort: [1:402:8] ICMP Destination Unreachable Port Unreachable [Classification: Misc activity] [Priority: 3]; (ICMP) 192.168.1.7 -> 192.168.1.1
	03-07-2010	02:57:35	Auth.Alert	127.0.0.1	snort: [1:402:8] ICMP Destination Unreachable Port Unreachable [Classification: Misc activity] [Priority: 3]; (ICMP) 192.168.1.7 -> 192.168.1.1
	03-07-2010	02:57:35	Auth.Alert	127.0.0.1	snort: [1:402:8] ICMP Destination Unreachable Port Unreachable [Classification: Misc activity] [Priority: 3]; (ICMP) 192.168.1.7 -> 192.168.1.1
	03-07-2010	02:56:43	Auth.Alert	127.0.0.1	snort: [1:408:5] ICMP Echo Reply [Classification: Misc activity] [Priority: 3]; (ICMP) 192.168.1.6 -> 192.168.1.2
	03-07-2010	02:56:43	Auth.Alert	127.0.0.1	snort: [1:466:5] ICMP L3retreiver Ping [Classification: Attempted Information Leak] [Priority: 2]; (ICMP) 192.168.1.2 -> 192.168.1.6
	03-07-2010	02:56:43	Auth.Alert	127.0.0.1	snort: [1:384:5] ICMP PING [Classification: Misc activity] [Priority: 3]; (ICMP) 192.168.1.2 -> 192.168.1.6
	03-07-2010	02:53:10	Auth.Alert	127.0.0.1	snort: [1:7250:8] NETBIOS SMB-DS srvsvc NetPathCanonicalize WriteAndX little endian overflow attempt [Classification: Attempted Administrator Privilege Gain] [Priority: 1]; (TCP) 192.168.1.7:57752 -> 192.168.1.6:445
	03-07-2010	02:53:09	Auth.Alert	127.0.0.1	snort: [1:7250:8] NETBIOS SMB-DS srvsvc NetPathCanonicalize WriteAndX little endian overflow attempt [Classification: Attempted Administrator Privilege Gain] [Priority: 1]; (TCP) 192.168.1.7:57752 -> 192.168.1.6:445

**Figure 19: Vulnerability Exploit**

The attack progress happens with vulnerability selected and each system goes through a cycle of forensics and anti forensics process. The diagram clearly shows the relation between forensics and anti forensics event happening at the same time. There are nearly 400 exploits in the



**Figure 20: Experimental Result as MES Diagram**

metasploit framework which can be deployed to any target in the network which demonstrates the fact that anti-forensic activity can always subvert forensics. The diagram in Figure 20 represents the forensic and anti-forensic activity at every stage of the attack as it progresses. This

diagram represents the experiment in logical way on whether anti-forensic activity is possible in a network.

#### **4.5 Evidence in Relation to AF**

The experiments in above section demonstrate the susceptibility of forensic efforts to anti-forensics attacks. The experiment in section 4.4.1 successfully demonstrated AF activity and so any evidence collected will have no legal value where AF activity is demonstrated. The experiment in section 4.4.2 demonstrated AF but the current procedures (MD5) detected the AF activity and the evidence collected can be considered as circumstantial evidence. In our experiment circumstantial evidence would be the image file from the system where AF is detected but the compromise was not successful. So the circumstance surrounding the investigation will determine the legal validity of evidence. The network AF experiment showed partial AF detection as it failed to identify deleted logs but successfully detected intrusion. The evidence collected in this experiment will be evidence of 'intent' as the intention of the person compromising a system is usually a matter to be determined by inference.

The current method of detection of AF is not rigorous and systematic. There is a need to detect anti-forensic activity to clarify the doubt surrounding the evidence and this research proposed one such model.



## 4.6 Summary

In this chapter, the need for a validation (meta-forensic) model was established. This was done in three main sections. In the first section, the testing framework was explained going through the NIST CFTT and Brian Carrier's DFTT. A combination method between the two was reached. The second section demonstrated that AF is possible in individual and networked system. The combination method used tools like secure remove, encase and the coroners toolkit which themselves lacked functionality and failed to detect AF. The last section addressed the need for a validation model looking at sort. The tools installed in the system provided useful insight of usability, status and features but lacked logical functionality to validate any evidence tampering. The experiment in section 4.4.2 demonstrated lack of functionality of forensic tools. The experiment also demonstrated the level of confidence in srm and encases tools. This confidence is a perceived confidence by use of the tools for a given situation.

The experiments also looked at evidence in legal perspective and its relation to AF. If it can be shown that when a success AF activity has taken place the evidence is court becomes invalid. The later part of experiments shows identification of vulnerabilities on a system requires only the knowledge of the tools, it does not require the knowledge of Internet protocols, programming language or anything else. Compromise of such systems is practically possible, because the body of knowledge to be searched is freely available. Also this knowledge is updated constantly. Vulnerabilities when exploited successfully question the validity of evidence. To address this issue we propose the meta-forensic techniques to validate the data. The rest of this thesis will attempt to address this issue.

## CHAPTER 5: THE META-FORENSIC MODEL

### 5.1 Classification

The experiment in the previous chapter attempted to demonstrate the threat to digital evidence integrity by anti-forensic activity. The research problem is intended as a statement of the compromise to digital evidence irrespective of detection of an anti-forensic attack. The point is that the validity of digital forensic evidence is contingent upon meta-forensic methods (meta-forensic is considered to be counter-anti-forensic). The vulnerabilities to anti-forensic attacks fall into one of four categories as shown in Table 4. Rodgers [76] classified anti-forensics into:

<b>Data hiding (D)</b> <ul style="list-style-type: none"><li>• Root kits (<b>D-r</b>)</li><li>• Encryption (<b>D-e</b>)</li><li>• Steganography (<b>D-s</b>)</li></ul>	<b>Obfuscation (OB)</b> <ul style="list-style-type: none"><li>• Log cleaners (<b>OB-l</b>)</li><li>• Spoofing (<b>OB-p</b>)</li><li>• Misinformation (<b>OB-m</b>)</li><li>• Zombied accounts (<b>OB-z</b>)</li><li>• Trojan commands (<b>OB-t</b>)</li></ul>
<b>Artifact wiping (W)</b> <ul style="list-style-type: none"><li>• Disk cleaner (<b>W-d</b>)</li><li>• Free space and memory cleaners (<b>W-c</b>)</li><li>• Prophylactic (<b>W-p</b>)</li></ul>	<b>Attacks against the tools(AT)</b> <ul style="list-style-type: none"><li>• File signature altering (<b>AT-a</b>)</li><li>• Hash fooling (<b>AT-h</b>)</li><li>• Nested directories (<b>AT-n</b>)</li></ul>

Table 4: Anti-Forensic Categories

The anti-forensic categories are labelled as D, OB, W and AT for the purposes of the research presented in this thesis. We now define additional subcategories as shown in Table 5. A

nomenclature is developed to group the anti-forensic sub categories so they can be easily identified.

<p>Root kits (<b>D-r</b>)</p> <ul style="list-style-type: none"> <li>• Persistent Root kits (<b>D-r-pr</b>)</li> <li>• Memory-Based Root kits (<b>D-r-mr</b>)</li> <li>• User-mode Root kits (<b>D-r-ur</b>)</li> <li>• Kernel-mode Root kits (<b>D-r-kr</b>)</li> </ul> <p>Encryption (<b>D-e</b>)</p> <ul style="list-style-type: none"> <li>• Symmetric encryption (<b>D-e-s</b>)</li> <li>• Asymmetric encryption (<b>D-e-as</b>)</li> </ul>
<p>Steganography (<b>D-s</b>)</p> <ul style="list-style-type: none"> <li>• Data hiding (<b>D-s-dh</b>)</li> <li>• Document Marking (<b>D-s-dm</b>)</li> <li>• Watermarking (<b>D-s-wm</b>)</li> <li>• Fingerprinting (<b>D-s-fm</b>)</li> </ul>

Table 5: Anti-Forensic Sub Categories

Capital letters are used for main category. Sub-categories are denoted using hyphen and lower case letters.

There are thousands of identified virus Trojans and security compromises. These have already been categorised by antivirus vendors [77, 78, 79]. All of these fall under one of the above categories. The classification can always be extended to new categories. We use Rogers’s classification and extend it to sub categories because model construction requires computation and the classification helps us to analyse results at every stage of decision making. Classification can show AF activity at higher level which can apply to a set of categories when classified and can be drilled down to sub categories if required to better understand the AF attacks. We require a baseline to start our model and classification forms a baseline for it.

Now that anti-forensic activity has been further classified, this chapter will begin looking at the components for a meta-forensic process.

## **5.2 Decision Tree**

To achieve evidence validation using meta-forensics, evidence is validated at each step of the evidence gathering process using a decision tree. Identifying anti-forensic behaviour is important to determine if evidence is compromised. Identifying the chain of events leading up to the anti-forensic behaviour and then modelling relationships between these events is required to demonstrate if AF activity had taken place. This is not sufficient to validate evidence so we extend this by proposing the decision tree concept.

There may be problems validating evidence using standard probability theory and binary logic because of the range of questions that would have to be answered before the analysis could even begin. For example

Compromising security by replacing a file is classified as a hack and no attempt has been made to suppress the evidence. So by our definition in section 1.6 we do not classify this as anti-forensic activity as no attempts to conceal the evidence had taken place. This leads us to what is the probability the examiner will detect this compromise during forensic investigation? If the examiner failed to detect this hack a new condition arises outside our AF definition. In this scenario the concept of decision tree is helpful. The concept of a hack can be expressed as an equation in binary form. The binary value will only have two sets of condition and to state our

decision tree a multi valued approach is required. AF activity can be expressed as qualitative and quantitative way. Hence the AF concept cannot easily be defined by classical logic so combination of classic logic and fuzzy logic is used to develop a mathematical model of AF activity and to analyse the issues associated with the application of the model.

Anti-forensic activity should be thought of as an object with various attributes [80]. As we move into validating its attributes, the decision-making process becomes problematic. In classical logic, the law of bivalence states that every proposition is either true or false. But anti-forensic detection requires the measurement of intermediate degrees of truth .In section 1.1 we discussed the presence of tools can establish AF which is not true or false as these tools are also used for privacy management and can be part of normal system operations. Thus, meta-forensics needs to employ fuzzy sets and fuzzy logic [81].

Using As shown in Figure 21, using the labelling defined in tables from section 4.1, the top of the tree represents the anti-forensic categories and the child trees represent subcategories which map to threat analysis. The subcategories are further divided into specific known vulnerabilities using Threat Assessment Methodology (TAME) [82]. Our model is extensible, new nodes can be added as and when new forms of tools or methods are discovered. The number of nodes and sub-nodes under each category is split into logical levels to make it easy to interpret the results.

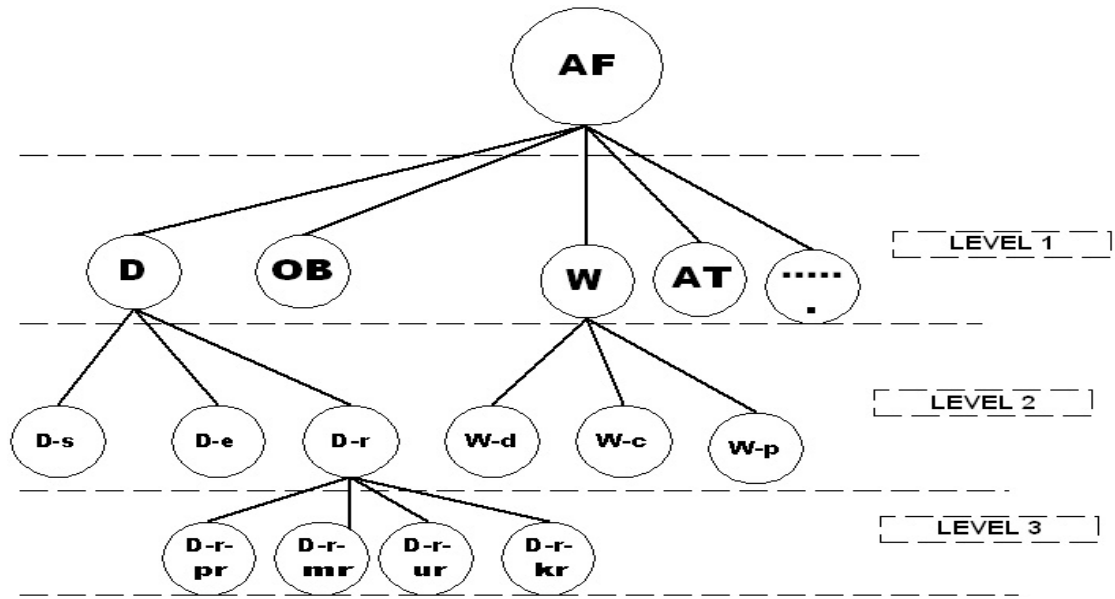


Figure 21: Anti-forensic activity tree

### 5.2.1 Level Depth

From the figure above we can see that there are three levels at which the anti-forensic subcategories are classified. We refer to this as depth. Level 1 is the top classification of anti-forensic categories. Level 2 and level 3 are subcategories. For anti-forensic activity to be in a level and category it has to satisfy certain conditions. One can question what if a tools or anti-forensic attacks fall under both the categories? For example disk cleaner is hidden inside a image file which wipes areas of a disk .Do we classify it under Data Hiding (D) or Artifact wiping (W). In this thesis research we classify it according to its order of discovery.

From Vulnerability Tree Model framework section 3.16 vulnerabilities were classified as a result of the relationship between vulnerabilities and the techniques used to exploit them by using codes was demonstrated in section 4.4.4. A malicious code attaches itself to a program or file enabling it to spread from one computer to another. The characteristics of this code is almost always that it attaches itself to an executable file, which means the it may exist on a computer but it actually cannot infect the computer unless a program is run or open. This characteristic is well defined for a computer virus. So we classify this under OB as level 2 data. If this virus has a malicious pay load and is a known Trojan we can classify it under level 3.

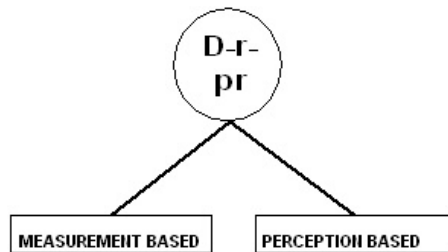
### **5.3 The Measurement and Perception Based Information**

The Measurement Based Information (MB) is derived from known sources and previous knowledge. As the name suggests, measurement based information is obtained by measurement and has a quantitative value. Various factors influence the measurement based information such as hardware, software, network connectivity, binaries, ports, default install location etc. The MB value is derived from a consolidation of individual measurements, the details of which were shown below.

Examples of Measurement Based Information include:

- Number of malicious payloads.
- Number of corrupted files.
- The number of files that have had their ordering changed according to the timestamp on the file.

For a given node MB and PB values are required components of meta-forensic model to detect AF. Shown below in Figure 22 is the classification of node information.



**Figure 22: Node information classification**

Perception Based information (PB) is information perceived by the investigator or a computer security officer. This value is determined from the analysis of similar previously investigated cases. The perceived value is an objective measurement and it is unlikely that any two cases will be exactly alike but the basic similarities always exist. However, we try to determine a meaningful value for PB information from the information obtained from past experience. The PB value is based on the difference between the normal system operation and the perceived change. The values contributing to PB information are qualitative. To determine the PB value one should understand a normal system operation. From our experiments in the previous chapter we consider normal system operation as a clean build machine before the forensic and anti forensic tools are deployed. Examples of Perception Based Information include:

- Considerations of hardware, operating system and software used in system.
- Considerations of Forensic tools (FTK, Encase) used to detect AF activity compared to intrusion prevention tools (firewalls, anti-virus software) used to prevent the AF activity.
- Considerations of the access control mechanism used in system before and after the system after the incident.



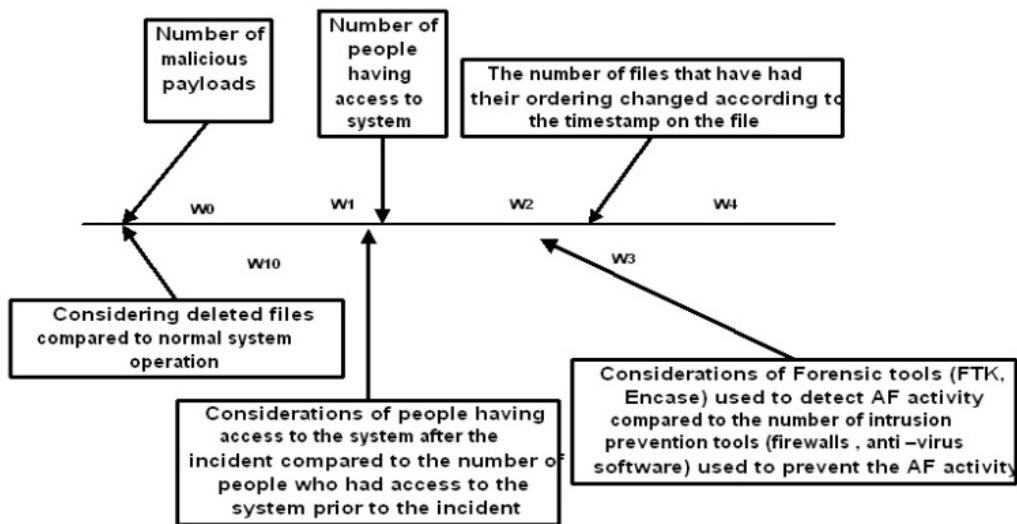
The considerations may seem like a quantitative measure but we consider reliability, usability of the tools, hardware, and software and access control mechanism by the confidence level it produced during our experiment in chapter 4.

#### **5.4 Assigning Weights**

The classified information above (MB and PB) is based on quantitative and qualitative measures respectively. To obtain results we use reasoning approaches discussed in Section 3.17. The result is obtained by assigning the weights to the events. The conditions that an event must satisfy to justify its assigned value are illustrated in Table 5 below. The weight of an individual event condition is given as a number between 0 and 10 where 0 is clearly defined as no access to the system and 10 represents a complete change in the system. Our logic is not just limited to choose 0 to 10 but can take values of 0 to 100 or 0 to 1000 as this is only a logical representation of the facts. The weight given to an event is logically related to the capabilities, qualities and the resources that it already possesses or can acquire in the future. Assigning a weight to an event condition is a one of process which can be later fed into a large database to be integrated into high level working model.

Figure 23 shows how to assign weights. Before assigning the weights the events are listed across a weight line for each node and the person assigning the weights starts the assignment process by considering the conditions of the events. The conditions are logical reasoning which was discussed in literature. The conditions determine the weights given to the events. The events can

be assigned individual weights or can be paired with other events and the pair assigned a weight. An example of such a pairing is the number of malicious payloads and consideration of forensic tools used compared to the number of intrusion prevention tools used. The conditions the events satisfy, such as the number of system log files being deleted every time a computer boots up is comparable to normal system operation and the number of people having access to the system prior to the incident is average, determine the weight assigned. An overall weighting is given to the combined effect of the events and the combination of the events weights.



**Figure 23: Assigning weights for a node**

We now justify why we arrive at a particular value for a particular event. Table 5 was decided upon by reasoning as it involves moving from a set of specific facts to a general conclusion. The possibilities which are included in table are to illustrate the logic where the weights can only take integer values. As shown below, measurement and/or perception based values can determine the weight assigned to an event condition.

To justify the weight assigned for a given node, we consider the analysed system node as an Extended Finite State Machine. So for the system state to change a set of triggers need to happen and the trigger conditions are events shown in Table 5. Both the quantitative and qualitative aspects have been considered so that justification of an event being low, medium or high can be explained.

If condition {No malicious payloads, No access to system. No change in file structure} is true then No change in system state.

If condition {No malicious payloads, No access to system. No change in file structure} is false then the conclusion is system state is changed.

So the logical Weight value for a No state is 0 and for a Yes state can be between 1 and 10.

If condition {All Log files deleted, All security permissions changed, All access to system changed} is true then the assigned logical Weight value is 10.

Now to assign a value between 1 to 9 we backtrack and analyse what caused the incident? This leads us to approach 3 sections 3.17.3 on literature, attack trees and Vulnerability tree reasoning where we traverse down the tree to understand the event occurrence. The event occurrence can be one or more of the following:

Use new Method OR Use Existing Method

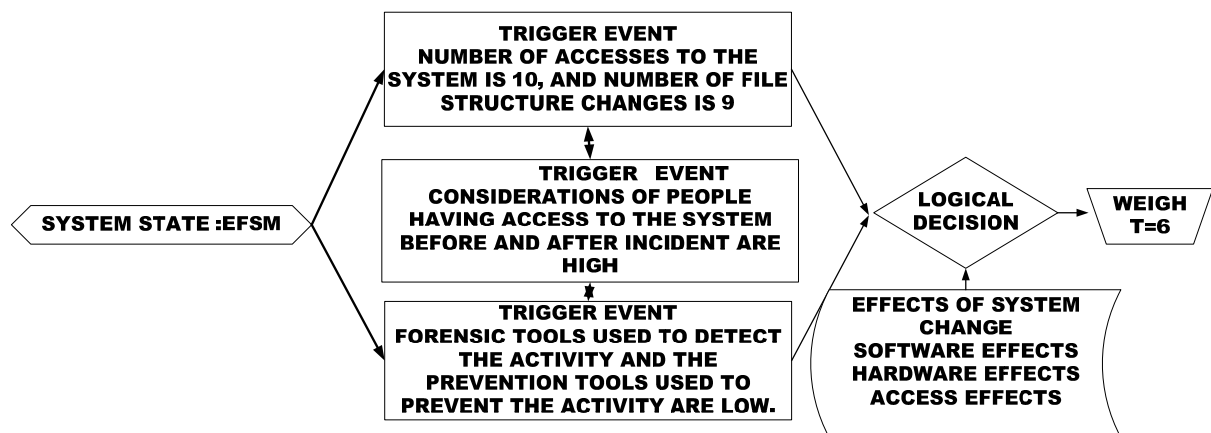
Develop own attack codes AND, OR Obtain Attack codes from Target person.

The above conditions are either possible or difficult. This leads us to approach 4 sections 3.17.4 to question the logic in approach 3 Why/Because new method was used instead of existing method and the cause and effect of it.

Causes {what caused the node incident to happen? software, hardware} Effects of it in system.

The above reasoning approach determines the assigned weight. It is possible to code the weights and descriptions by constructing a WB graph approach discussed in literature. A WB graph is shown in Figure 24. To justify the assigned weight for a given event we summarize the above logic.

- System state – Extended Finite State Machine.
- Trigger Events – The MB PB value.
- Logical Decision – Justification for assigning weight.
- Effects – Software, Hardware, and Access control effects.



**Figure 24: WB reasoning for a node**

The evidence is in the form of characteristics which are exhibited in the form of events. The cause of this event leads us to the effects and the logical combination of cause effect ,why/ because, an incident happens determines the weight value. A classical example will be to mark

an oral presentation for group of students. The judges determine the scores which is a numeric value based on the performance of the candidates.

WEIGHTS VALUE W	EVENT CONDITIONS
0	No malicious payloads, No access to system. No change in file structure.  Considerations of deleted files compared to normal system operation are low.
1	Number of malicious payloads is 2. Number of accesses to system is 3, and  Number of file structure changes is 1. Considerations of deleted files compared to normal system operation low.
2	Number of malicious payloads is 3. Number of files that had their ordering changed is 5. Considerations of people having access to the system before and after incident are low.
3	Number of malicious payloads is 6. Number of files that had their ordering changed is 7. Considerations of people having access to the system before and after incident are low.
4	Number of malicious payloads is 7. Number of files that had their ordering changed is 6. Considerations of people having access to the system before and after incident are high.

5	<p>No malicious payloads, No access to system. No change in file structure.</p> <p>Considerations of deleted files compared to normal system operation are high.</p> <p>Considerations of forensic tools used to detect the activity are low and the prevention tools used to prevent the activity is low.</p>
6	<p>Number of accesses to the system is 10, and Number of file structure changes is 9.</p> <p>Considerations of forensic tools used to detect the activity and the prevention tools used to prevent the activity are low. Considerations of people having access to the system before and after incident are high.</p>
7	<p>Number of accesses to the system is 12, and Number of file structure changes is 11.</p> <p>Considerations of forensic tools used to detect the activity and the prevention tools used to prevent the activity are low.</p>
8	<p>Number of access to system is 18, and Number of file structure changes is 15.</p> <p>Considerations of forensic tools used to detect the activity and the prevention tools used to prevent the activity are low.</p>
9	<p>All Log files deleted. All security permissions changed. Considerations of deleted files compared to normal system operation are high.</p>
10	<p>All Log files deleted. All security permissions changed. Considerations of deleted files compared to normal system operation are high. Considerations of forensic tools used to detect the activity are low and the prevention tools used to prevent the activity are low.</p>

Table 5: Logically assigning weights

The outcome of the performance is to impress the judge but one or more bits during performance (the characteristics or properties) influence the judge to arrive at a conclusion. In short they are following a WB analysis to arrive at a value. Similarly the events surrounding the anti-forensic activity and the cause effect interrelationship between the events determines the weight. If two different examiners are asked to mark this weight the result might vary. However what we are trying to establish is a method to determine the weights. Once the method is worked out we can create a standard for known anti-forensic activity. The same concept is used above to determine the value. The presence of evidence linking to the file determines the value. One can argue that the data provides us with evidence for the numbers and the evidence is exhibited in form of data but the question of detecting unknown AF or validating the data is only possible when represented in numbers and logically analysed. Also during investigation this process can quickly guide a forensic investigation to the suspected AF activity. This process can be automated using an algorithm which can be programmed in any high level language.

```

<WEIGHT VALUE NODEID="eg:D-r-pr">
  <COUNTERFACTUAL REASONING>
    LOAD MB, PB
      <Approach 1>
        EFSM, Backtracking
          <Result>
<CONFIDENCE>
      <Low> </Low>
      <Medium></Medium>
      <High></High>
</CONFIDENCE>
  </Result>

```

</Approach 1>

<Approach 2>

Representation in MES

<Result>

<ATTACK PROGRESSION>

<Time>

</Time>

</ATTACK PROGRESSION>

</Result>

</Approach 2>

<Approach3>

Vulnerability tree reasoning

</Possible>

YES, NO

</Possible>

</Difficult>

YES, NO

</Difficult>

<Result>

<CONFIDENCE>

<Low> </Low>

<Medium></Medium>

<High></High>

</CONFIDENCE>

</Result>

</Approach3>

<Approach4>

Causes----- Nodes (AND, OR) ----Effects

</Approach4>

<WEIGHVALUE> Integer </WEIGHTVALUE>



</COUNTERFACTUAL REASONING>

</WEIGHT VALUE NODEID>

#### **5.4.1 Node Value after assigning weights**

Once the weights of all the events associated with a given node have been determined, the node value can be determined. This is an overall weighting based on the event weights. There is no restriction on the number of weights generated from specific events as an event may influence the overall weighting of several node values. The more event values we have the error on the system minimises. The assigned weights are stored in SQL database and can be expanded for future use. The reasoning applied in this research to set weights might differ from person to person depending on their knowledge and expertise in the subject. But this is a onetime activity and the more experts assigning the system weights can improve the system further. This area of data collection forms a new research area. Once the weights for a given node have been determined the sum of the weights gives us the node value.

*W = Weight Value*

$$\text{Node value for } D - r - pr = \sum_{i=1}^n W_i$$

The node value is used to determine a fuzzy value for the node which may be Low, Medium or High. The value assigned depends on number of weights contributing to the node value. If the maximum value a node can take is 100 in our opinion we then classify low as a node value

between 0 and 33, medium as a node value between 33 and 66, and high as a node value between 66 and 100. In future when the system expands if the maximum value a node can take is 1000 in our opinion we then classify low as a node value between 0 and 333, medium as a node value between 333 and 666, and high as a node value between 666 and 1000.

#### 5.4.2 Fuzzy Universal Set

The above process is explained as follows.

**Node Leaf ----- MB, PB Value -----Weight Value (W) -----Low, Medium, High**

After deriving the fuzzy value (Low, Medium, High) of each node in the tree the relationships between the nodes are determined using fuzzy logic. We denote a vector whose elements are the weights assigned to the actual conditions of the events being used to analyse a particular system.

Define a universal fuzzy set:  $S =$  set of all possible vectors  $s$  for a particular analysed system. In our case, every instance of  $s$  has to result in one of three fuzzy values, namely, low (L), medium (M), or high (H) being assigned to each node of the analysed system. If in a particular case we have only two nodes,  $D-r-pr$  and  $D-r-mr$ .

$$\varphi L^{(D-r-pr)} = \{s \in \varphi(D-r-pr) \mid s \text{ generates a value Low}\}$$

$$\varphi M^{(D-r-pr)} = \{s \in \varphi(D-r-pr) \mid s \text{ generates a value Medium}\}$$

$$\varphi H^{(D-r-pr)} = \{s \in \varphi(D-r-pr) \mid s \text{ generates a value High}\}$$

$$\varphi L^{(D-r-mr)} = \{s \in \varphi(D-r-mr) \mid s \text{ generates a value Low}\}$$

$$\varphi M^{(D-r-mr)} = \{s \in \varphi(D-r-mr) \mid s \text{ generates a value Medium}\}$$

$$\varphi H^{(D-r-mr)} = \{s \in \varphi(D-r-mr) \mid s \text{ generates a value High}\}$$

Then in effect we have two sets  $\varphi(D-r-pr)$  and  $\varphi(D-r-mr)$ , to consider. We can partition these sets as above.

We have two nodes each taking one of the three possible fuzzy values of low, medium, or high as a result of the weighting and weight summing process. To describe the perception of anti-forensic activity on a node, we describe the perception of high, medium, and low, respectively, as being anti-forensic activity is very likely, likely, and unlikely respectively. Very likely, likely, and unlikely are defined linguistic variables in fuzzy logic [83]. This information can be represented as an ordered pair using Zadeh's principals [84, 85, and 86]. For the case considered we would have the description:

**AF activity: ((very likely, high activity), (likely, medium activity), (unlikely, low activity)).**

## 5.5 Operations

The above statements can be extended to fuzzy unions and intersections of nodes to highlight the anti-forensic activity on the system. The equations given are based on event weights described in table 5 section 5.4 and Figure 24 (WB reasoning for nodes). Within an event there are associated characteristic each of which provides a weighing event for making a decision and guide us towards the degree of confidence. We use a rating scale to compare between nodes. The

justification for using an AF rating scale is, in a situation surrounding AF activity the derived weighted node values already consist properties and characteristics of the node. So comparing the nodes against each other will give us an idea to describe how likely an AF event might have happened.

The scales used are:

Positive AF rating scale: likely, very likely

Negative AF rating scale: Unlikely

For example, perceived anti-forensic activity in a system with two nodes, D-r-pr and D-r-mr, could be described as follows:

If $\varphi_L^{(D-r-pr)} \cap \varphi_L^{(D-r-mr)}$	then Unlikely
If $\varphi_L^{(D-r-pr)} \cap \varphi_M^{(D-r-mr)}$	then Unlikely
If $\varphi_L^{(D-r-pr)} \cap \varphi_H^{(D-r-mr)}$	then Very likely
If $\varphi_M^{(D-r-pr)} \cap \varphi_L^{(D-r-mr)}$	then Unlikely
If $\varphi_M^{(D-r-pr)} \cap \varphi_M^{(D-r-mr)}$	then Likely
If $\varphi_M^{(D-r-pr)} \cap \varphi_H^{(D-r-mr)}$	then Very likely
If $\varphi_H^{(D-r-pr)} \cap \varphi_L^{(D-r-mr)}$	then Likely
If $\varphi_H^{(D-r-pr)} \cap \varphi_M^{(D-r-mr)}$	then Very likely
If $\varphi_H^{(D-r-pr)} \cap \varphi_H^{(D-r-mr)}$	then Very likely
If $\varphi_L^{(D-r-pr)} \cup \varphi_L^{(D-r-mr)}$	then Unlikely
If $\varphi_L^{(D-r-pr)} \cup \varphi_M^{(D-r-mr)}$	then Very likely
If $\varphi_L^{(D-r-pr)} \cup \varphi_H^{(D-r-mr)}$	then Very likely
If $\varphi_M^{(D-r-pr)} \cup \varphi_L^{(D-r-mr)}$	then Likely
If $\varphi_M^{(D-r-pr)} \cup \varphi_M^{(D-r-mr)}$	then Likely
If $\varphi_M^{(D-r-pr)} \cup \varphi_H^{(D-r-mr)}$	then Very likely
If $\varphi_H^{(D-r-pr)} \cup \varphi_L^{(D-r-mr)}$	then Likely
If $\varphi_H^{(D-r-pr)} \cup \varphi_M^{(D-r-mr)}$	then Very likely
If $\varphi_H^{(D-r-pr)} \cup \varphi_H^{(D-r-mr)}$	then Very likely

The union and intersection used here helps us to express the event occurrence. The intersections are combined by min and union by max operation and the outcome is expressed as unlikely, likely and very likely depending on low, medium and high activity. This operation does not strictly adhere to the set theory principals as we do not consider the null sets, but we use this as a mere form of expressing the event outcome.

$$MIN(\varphi L(D - r - pr), \varphi(D - r - mr)) \text{ then } Unlikely$$

$$MAX(\varphi L(D - r - pr), \varphi(D - r - mr)) \text{ then } Unlikely$$

## 5.6 Modelling the Anti-forensic Activity

An effective anti-forensic system should possess the following qualities:

1. The ability to translate the above steps into programmable steps.
2. A capacity for the process to be automated and minimum intervention by the investigator.
3. An ability to add new input variables for future analysis independently of the current MB and PB approach.
4. The flexibility to print the output results in any form for easy interpretation.
5. The ability to model multi-stage anti-forensics systems like anti-anti forensics systems.
6. The ability to translate between logged data and an actual event in a one-to-one fashion.
7. The system should be easily implemented using any practical methodology.

A working model designed using Matlab is shown in Figure 25 for a level 3 node. There are 18 membership rules in terms of the likelihood of fuzzy unions and intersections considered for this particular case which have been given in section 5.2.

In this case, every instance of  $s$  has to result in one of three fuzzy values, namely, low (L), medium (M), or high (H) being assigned to each node of the analysed system. We have four nodes,  $D-r-pr$ ,  $D-r-mr$ ,  $D-r-ur$  and  $D-r-kr$ . Then in effect we have four sets  $\varphi(D-r-pr)$ ,  $\varphi(D-r-mr)$ ,  $\varphi(D-r-ur)$ ,  $\varphi(D-r-kr)$ , to consider. We can partition these sets as follows:

$$\varphi L^{(D-r-pr)} = \{s \in \varphi(D-r-pr) \mid s \text{ generates a value Low}\}$$

$$\varphi M^{(D-r-pr)} = \{s \in \varphi(D-r-pr) \mid s \text{ generates a value Medium}\}$$

$$\varphi H^{(D-r-pr)} = \{s \in \varphi(D-r-pr) \mid s \text{ generates a value High}\}$$

$$\varphi L^{(D-r-mr)} = \{s \in \varphi(D-r-mr) \mid s \text{ generates a value Low}\}$$

$$\varphi M^{(D-r-mr)} = \{s \in \varphi(D-r-mr) \mid s \text{ generates a value Medium}\}$$

$$\varphi H^{(D-r-mr)} = \{s \in \varphi(D-r-mr) \mid s \text{ generates a value High}\}$$

$$\varphi L^{(D-r-ur)} = \{s \in \varphi(D-r-ur) \mid s \text{ generates a value Low}\}$$

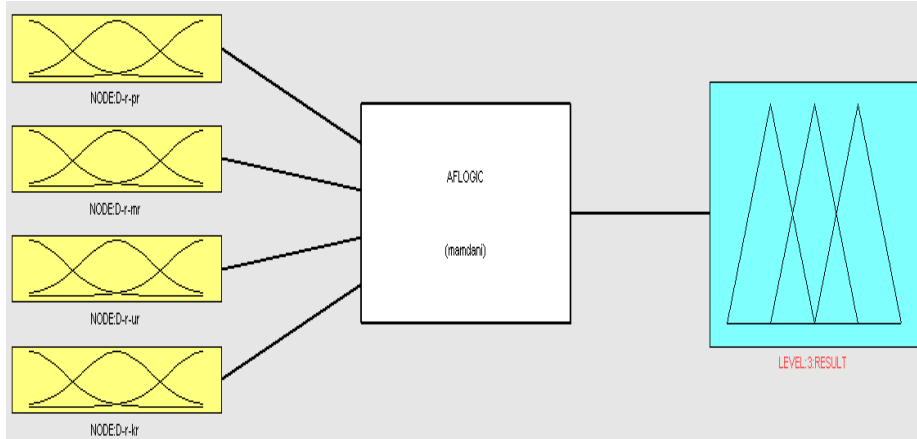
$$\varphi M^{(D-r-ur)} = \{s \in \varphi(D-r-ur) \mid s \text{ generates a value Medium}\}$$

$$\varphi H^{(D-r-ur)} = \{s \in \varphi(D-r-ur) \mid s \text{ generates a value High}\}$$

$$\varphi L^{(D-r-kr)} = \{s \in \varphi(D-r-kr) \mid s \text{ generates a value Low}\}$$

$$\varphi M^{(D-r-kr)} = \{s \in \varphi(D-r-kr) \mid s \text{ generates a value Medium}\}$$

$$\varphi H^{(D-r-kr)} = \{s \in \varphi(D-r-kr) \mid s \text{ generates a value High}\}$$



**Figure 25: Working model showing anti-forensic activity for a level 3 node**

If  $\varphi L^{(D-r-pr)} \cap \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi L^{(D-r-pr)} \cap \varphi M^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi L^{(D-r-pr)} \cap \varphi H^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi M^{(D-r-pr)} \cap \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi M^{(D-r-pr)} \cap \varphi M^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi M^{(D-r-pr)} \cap \varphi H^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi H^{(D-r-pr)} \cap \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi H^{(D-r-pr)} \cap \varphi M^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi H^{(D-r-pr)} \cap \varphi H^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi L^{(D-r-pr)} \cap \varphi M^{(D-r-mr)} \cap \varphi M^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi L^{(D-r-pr)} \cap \varphi H^{(D-r-mr)} \cap \varphi H^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Very Unlikely

If  $\varphi M^{(D-r-pr)} \cap \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi M^{(D-r-pr)} \cap \varphi M^{(D-r-mr)} \cap \varphi M^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi M^{(D-r-pr)} \cap \varphi H^{(D-r-mr)} \cap \varphi H^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Very Likely

If  $\varphi H^{(D-r-pr)} \cap \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi H^{(D-r-pr)} \cap \varphi M^{(D-r-mr)} \cap \varphi M^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi H^{(D-r-pr)} \cap \varphi H^{(D-r-mr)} \cap \varphi H^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Very Likely

If  $\varphi L^{(D-r-pr)} \cup \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi L^{(D-r-pr)} \cup \varphi M^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi L^{(D-r-pr)} \cup \varphi H^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi M^{(D-r-pr)} \cup \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi M^{(D-r-pr)} \cup \varphi M^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi M^{(D-r-pr)} \cup \varphi H^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cap \varphi L^{(D-r-kr)}$  then Very Likely

If  $\varphi L^{(D-r-pr)} \cup \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cup \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi L^{(D-r-pr)} \cup \varphi M^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cup \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi L^{(D-r-pr)} \cup \varphi H^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cup \varphi L^{(D-r-kr)}$  then Likely



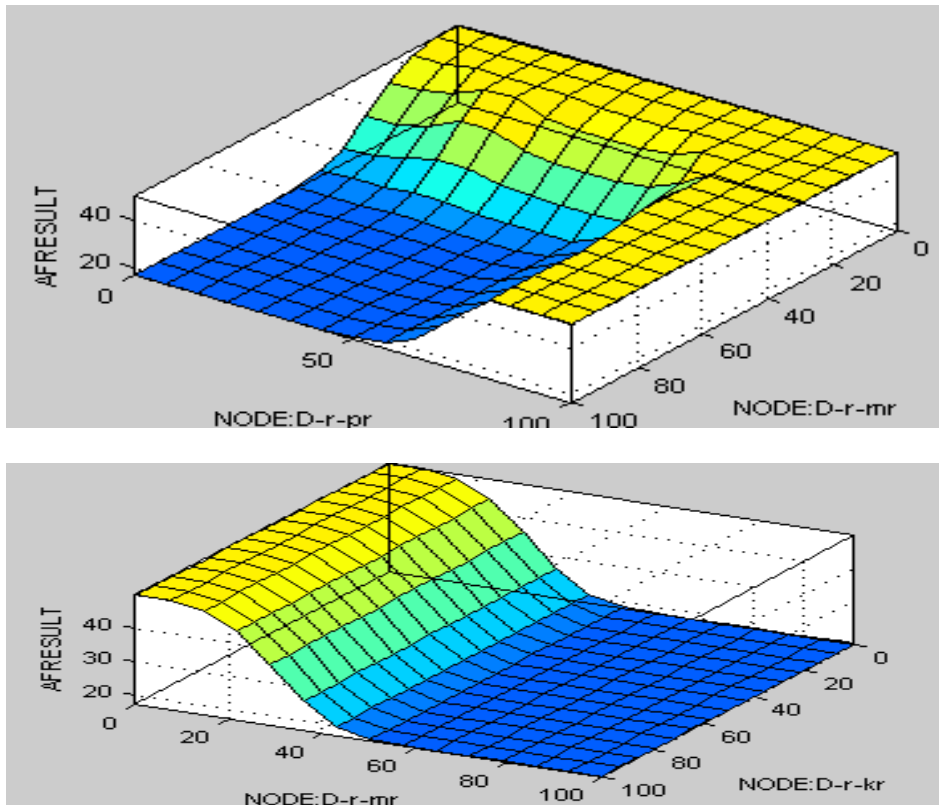
If  $\varphi M^{(D-r-pr)} \cup \varphi L^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cup \varphi L^{(D-r-kr)}$  then Unlikely

If  $\varphi M^{(D-r-pr)} \cup \varphi M^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cup \varphi L^{(D-r-kr)}$  then Likely

If  $\varphi M^{(D-r-pr)} \cup \varphi H^{(D-r-mr)} \cap \varphi L^{(D-r-ur)} \cup \varphi L^{(D-r-kr)}$  then Very Likely

The anti-forensic results are analysed for each logical level in the tree. As the node increases the combination between them become larger to interperate as equations .The above equation only shows some of the intersection between the nodes and a combination of union and intersection is possible. So in order to minimize the complexity the values (Low, Medium, High), union and intersection between nodes each node are analysed using a commonly used fuzzy inference technique known as the Mamdani's Method using 'min-max' operation. However, we are not restricted to using the Mamdani's Method and other fuzzy logic methods can also be used to derive the same result. The Mamdani's Method was adopted because of ease of use and implementation. The code used to design the system is shown in Appendix E.

For a given value of each node the anti-forensic activity graph varies between unlikely and likely for the input range of 0 and 100. The result is shown below as a surface graph which shows the likeliness of the anti-forensic activity on level 3 nodes.



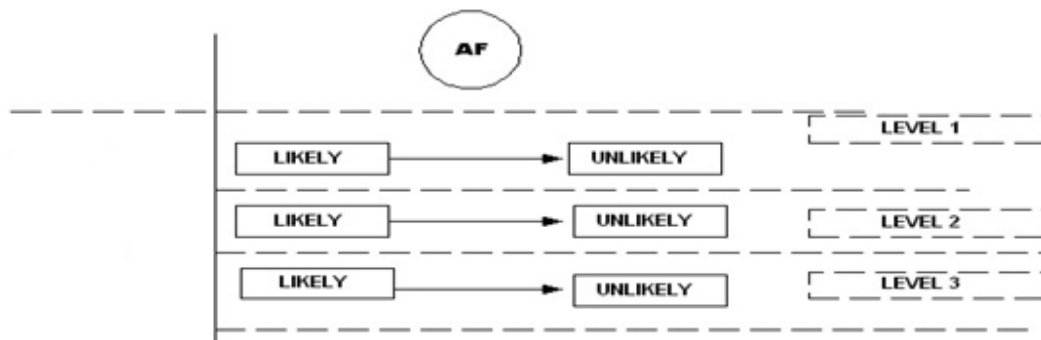
**Figure 26: AF activity result for level 3 nodes**

The dark blue colour in the first heat map shows AF activity for Node D-r-pr compared with D-r-mr .Nearly 50% of the AF activity is shown for the D-r-pr node. The lighter yellow shows no AF activity. Around 60% range there is a overlap which shows the comparison of both nodes taken together for the AF activity. Similarly in second heat map node D-r-kr shows more AF than node d-r-mr.

The same procedure is repeated for level 2 and level 1 nodes. MB and PB values for the parents are derived in the same manner as for child nodes, however, the weights differ according to the MB and PB used for reasoning. Hence, new node values have to be derived for different categories using similar logic and results analysed. This is discussed in an example in Section

5.7. The results obtained in level 1 and level 2 have less depth than those in level 3. The result of all the levels shows us the fuzzified output of anti-forensic activity of the entire system as shown in Figure 27. The results are spread between likely to unlikely forensic activity in the given system.

The representation shown on Figure 26 only helps forensic investigators to focus their attention more on areas showing AF activity. This will not be admissible in court of law as it's only a component to demonstrate the AF activity. The overall validity of evidence at all levels shown in Figure 28 and Figure 29 can be used to demonstrate the confidence in evidence gathered.



**Figure 27: Anti-forensic activity result for all nodes**

### 5.7 Model Application Case1

As an example, we take the case of a financial fraud committed via email against a large corporate will demonstrate how it works. The case is decomposed into the following sections in order to simplify presentation.

- Complaint received by investigation officer.

- Undercover operation.
- Meta-forensic approach taken.
- Evidence gathering and analysis of case.
- Result and conclusion.

The system administrator received complaints from a number of users about spam email from a person who claimed that their paid subscription will be free for life if a onetime payment of an amount is made in cash in a plain envelope at a designated place. The email claimed that the user can confirm their subscription details online and via customer services before dropping off the cash and only make a payment after the system is updated for lifetime online membership. It also claimed if the money is not received as said the system will revert back to original state and everyone involved should pretend that the incident never happened. As the case involved someone's claim to change member's subscription without authorisation it was decided to investigate the claim.

The case unfolded over the course of several days and the system administrators started tracking the network and IP of the email. The headers of the original emails were analysed. It was clear at this stage that the email had come from the internal network via a newly created hotmail account. The internal IP address of the originator computer was detected in email headers and the hotmail profile creation date matched the date the incident was reported. All traffic from the subnets concerned was monitored. The number of people visiting hotmail from the subnet range was tracked for last six months from access logs in network. An undercover operation was planned and one of the users was asked to respond to the email. As the email communication started the

username and computer which was used to send the email reply was identified. Forensic investigators were called in and the evidence from the computer was gathered.

The evidence gathered was in form of log files from the workstation, server, and network showing the activity of the user. At this stage the owner of the computer was confronted with the evidence but did not accept responsibility for this incident and claimed that the evidence had been fabricated. His defence was someone had been hiding a malicious program which caused this incident. An AF approach was applied to strengthen the evidence gathered at various stages of investigation. It was decided to look for data hiding and wiping software which would have compromised user accounts. As this stage it did not limit us to run other AF detection but as the claim was only for data hiding and wiping it was decided to just try the two node approach .

From the anti-forensic activity tree shown in Figure 21 Section 5.2 the data hiding (D) and artifact wiping (W) node was analysed for this case. The perception based (PB) and measurement based (MB) inputs for the data hiding nodes are as follows:

Measurement Based Information (MB):

- New directories created not associated with any programs= 12.
- Backdoor entry programs= 0.
- Corrupted access log files =1.
- Files that have had their ordering changed according to the timestamp on the file =1.

Perception Based Information (PB):

- Consideration of hits to security/hack tools website compared to normal hit during the same period.
- Consideration of Temp files deleted just after the email correspondence was high compared to normal system operation.
- Consideration of access to membership data using admin account was high compared to normal system operation.
- Consideration of access to the email system was high compared to normal system operation.

We now have 8 conditions to be analysed for a data hiding node. The same MB and PB information can be used for level 2 and level 3 for data hiding node and sub nodes.

The perception based (PB) and measurement based (MB) input for the artifact wiping nodes are as follows:

#### Measurement Based Information (MB):

- Deleted system files from an operating system=0.
- File system signatures to determine if the file system was wiped=1.
- Corrupted access log files=2.
- Free bytes available compared to normal system resource=12gb.

#### Perception Based Information (PB):

- Consideration of types of disk wiping tools used compared to other software installed.
- Consideration of Temp files deleted just after the email correspondence was high compared to normal system operation.

- Consideration of deletion of membership data logs using admin account immediately after the incident.
- Consideration of free disk space in system was high compared to normal system operation.
- Consideration of cpu and memory usage of system was high compared to normal system operation.

We now have 9 conditions to be analysed for an artifact wiping node. We analyse the MB and PB information and we derive the weights for the nodes at each level. This is shown in Table below. The sum of all individual weights determines the weight of any particular node. The node value can be Low, Medium or High. To justify the weight assignment we follow the logical approach discussed in section 5.4 and literature review.

<WEIGHT VALUE NODEID = D-r-pr>

<COUNTERFACTUAL REASONING>

LOAD MB, PB (Number of corrupted access log files =1, considering access to membership data using admin account during that period was high compared to normal system operation)

<Approach 1>

EFSM, Backtracking

<Result>

<CONFIDENCE access to membership data >

<Low> NO</Low>

<Medium>NO </Medium>

<High> YES </High>

</CONFIDENCE>

</Result>

</Approach 1>

<Approach 2>

Representation in MES (The time it took for the event to happen)

```

    <Result>
<ATTACK PROGRESSION>
    <Time> 2 Seconds</Time>
</ATTACK PROGRESSION>
    </Result>
    </Approach 2>
    <Approach3>
        Vulnerability tree reasoning
        </Possible>
        New method used NO
        </Possible>
        </Possible>
        Existing method used YES
        </Possible>
        </Difficult> NO</Difficult>
    <Result>
<CONFIDENCE>
    <Low> NO</Low>
    <Medium>NO</Medium>
    <High>YES</High>
</CONFIDENCE>
    </Result>
    </Approach3>
    <Approach4>
Software error, Bug in system, Configuration issues----- Nodes (AND, OR) ----Log File Change
    </Approach4>
<WEIGHVALUE> "1" </WEIGHTVALUE>

```

</COUNTERFACTUAL REASONING>

</WEIGHT VALUE = D-r-pr >

We use similar reasoning to arrive at a weight value for both the Nodes for various events.



$D - r$ - pr $= \sum_{i=1}^n W_i$	$D - r$ - mr $= \sum_{i=1}^n W_i$	$D - r$ - ur $= \sum_{i=1}^n W_i$	$D - r$ - kr $= \sum_{i=1}^n W_i$	EVENTS
0	0	0	0	Number of backdoor entry programs= 0 and considering Number of deleted system files from an operating system=0
1	2	1	0	Number of corrupted access log files =1 considering access to membership data using admin account during that period was high compared to normal system operation
2	1	2	1	Number of new directories created not associated with any programs= 12 and considering hits to security/hack tools website compared to normal hit during the same period.
1	1	2	1	Number of new directories created not associated with any programs= 12 considering access to the email system was high compared to normal system operation and considering access to membership data using admin account was high compared to normal system operation
2	2	2	0	Number of files that have had their ordering changed according to the timestamp on the file =1 and considering access to the email system was high compared to normal system operation.
6	6	7	2	Final Value

Table 6: Logically assigning weights for data hiding

$w - d = \sum_{i=1}^n W_i$	$w - c = \sum_{i=1}^n W_i$	$w - p = \sum_{i=1}^n W_i$	EVENTS
0	0	0	Number of deleted system files from an operating system=0 considering Temp files deleted just after the email.
0	1	1	Number of file system signatures to determine the file system was wiped=1 considering the cpu and memory usage of system was high compared to normal system operation.
2	1	3	Number of corrupted access log files=2 considering deletion of membership data logs using admin account immediately after the incident.
1	1	4	Number of free bytes available compared to normal system resource=12gb considering free disk space in system was high compared to normal system operation.
3	3	8	Final Value

Table 7: Logically assigning weights for artifact wiping

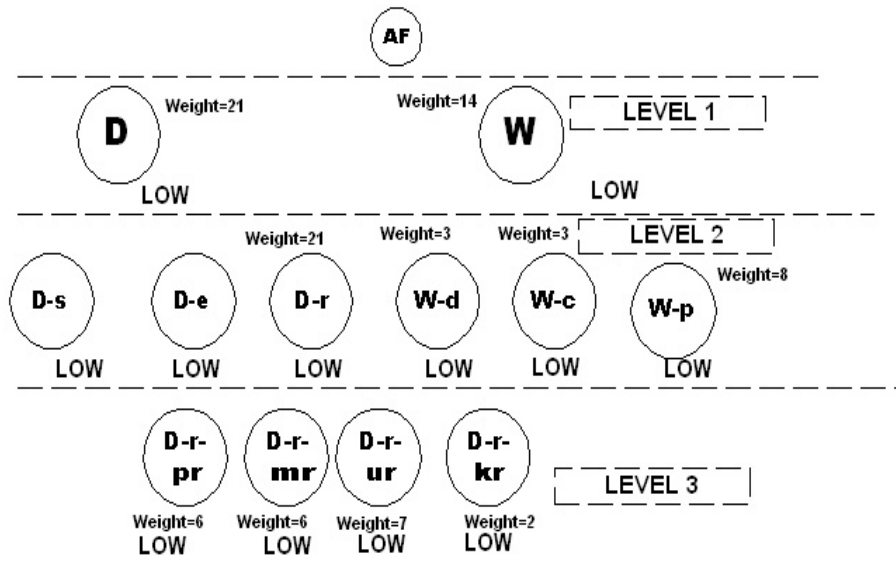
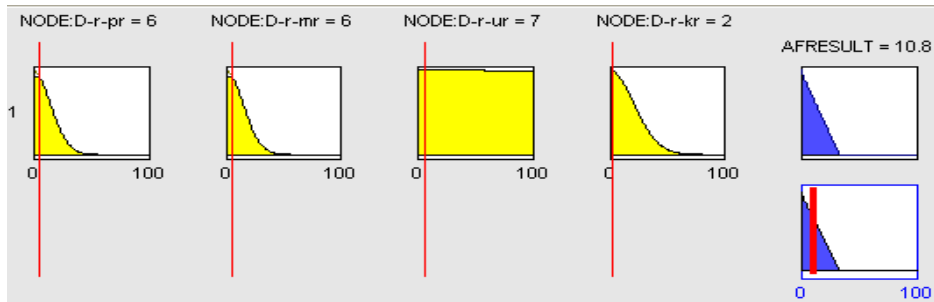


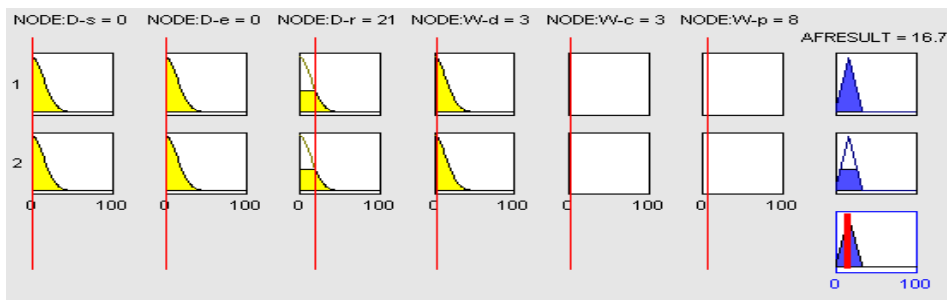
Figure 28: Reasoning Results

To determine the anti-forensic activity on the system the reasoning results are run using the analyser as indicated in Fig section 4.6. The output of AF the analyser for level 1, level 2 and level 3 are shown in Figure 32.

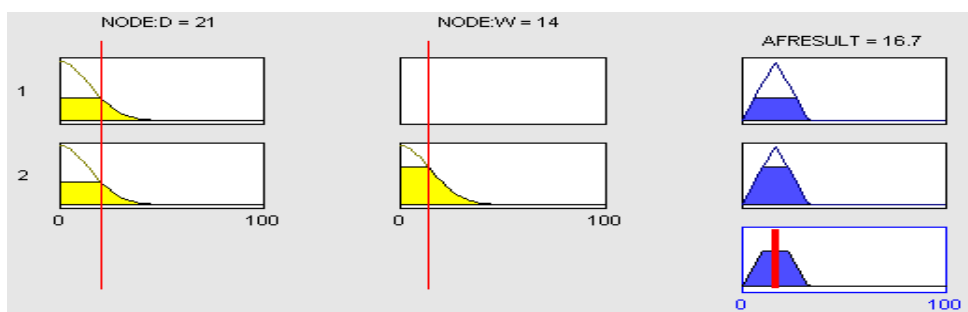
Level 3



Level 2



Level 1

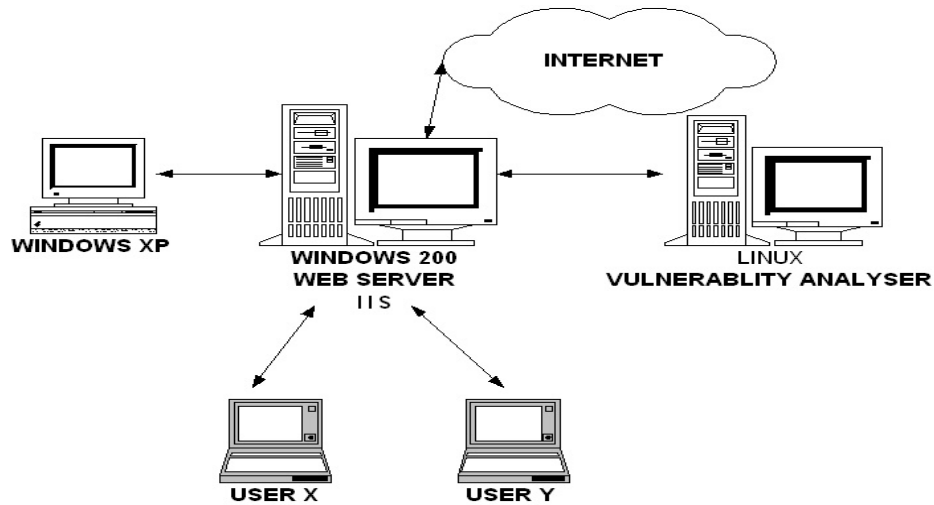


**Figure 29: Anti-forensic activity result for level 1, level 2 and level 3**

The result clearly shows the anti-forensic activity on all the nodes at all levels. From the result of all the three levels, the degree of anti-forensic activity was not significantly higher and so did not do enough to justify the claims made by the owner of the computer. At the time of investigation we used the above method to detect AF. Since the above method is not recognised as a standard system in courts to validate evidence it was only used to guide examiner to look at any possible AF activity in system. The case was tried in court and resulted in conviction using existing techniques.

## **5.8 Model Application Case 2**

The model is divided up into various sub-groups and components and each component is described in detail. The overall high level model is presented at the end of this chapter. The experimental setup consist of network which runs various services like web server, database server, IDS, IPS, in a networked environment. The VMware infrastructure used in chapter 4 is used for this experiment. A windows 2000 server is used as a web server. The web server hosts an externally facing website. The actor in this role is the system/network administrator managing the system. Other actors are defined as and when required. The architecture for application of meta-forensic model is shown in Figure 30.



**Figure 30: Architecture for application of meta-forensic model**

There are two users X & Y who are responsible for editing the webpage's. The system is designed such that only one user can edit the webpage as the page is locked preventing multiple edits.

### 5.8.1 The Issue

X claims that confidential organisation data was published on the web for few minutes when Y edited the pages, but Y denies any involvement. So the dispute goes to the forensic investigator to investigate this issue. The investigator finds out that the last two logons were Y's logon. The access log on the web server clearly shows people visiting the webpage. User X's and Y's IP addresses are registered in access logs. The system admin makes an image copy of the access logs of the websites. This follows the generic traditional forensic investigation procedure.

- Identification-----System Identified to be investigated.

- Preparation -----Log files Identified.
- Preservation-----Image Copy Saved.
- Analysis----- Log File Analysis
- Presentation & Conclusion -----Y edited the page

The forensic investigator reasons that the last two usernames present in the access log files are user Y's so he has edited the system. The IP address of X in the web access log proves that X has indeed visited the page and seen the information being available for some time. Therefore, it must be the case that Y has edited the system and published the confidential data.

This evidence statement given above is true but cannot be completely accepted as true evidential statement because it has not been subjected to a rigorous validation process and experiment in previous sections identifies this issue.

### **5.8.2 Meta-forensic Approach**

Since the whole evidence was based on the log files it is decided to start our analysis by looking for obfuscation method's and sub category for any evidence in system.

The perception based (PB) and measurement based (MB) input for the obfuscation nodes are as follows:

LEVEL 3:

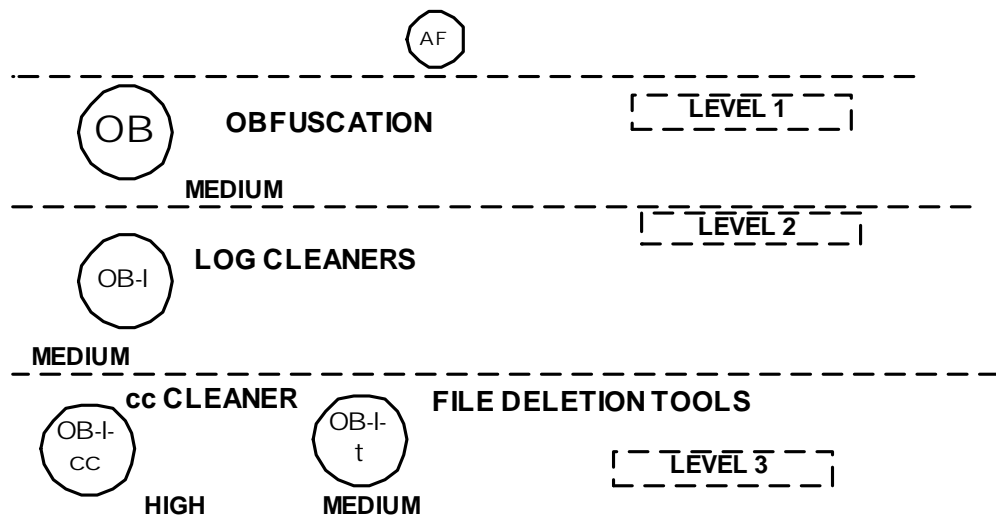
Measurement Based Information (MB):

- Number of deleted files from log files directory from time of incident to time of analysis=45.
- Number of file system signatures to determine if the file system was wiped=1.

- Number of corrupted access log files=7.
- Number of intrusion attempts during that period =49.
- Number of backdoor entry programs= 2

Perception Based Information (PB):

- Consideration of tools used for detection.
- Consideration of Temp files deleted just after the reported incident type was high compared to normal system operation.
- Consideration of antivirus software running during the incident.
- Consideration of vulnerability in the system.



**Figure 31 Obfuscation method's sub category**

By using logical reasoning for one of the events

<WEIGHT VALUE NODEID = OB-l-cc>

<COUNTERFACTUAL REASONING>

LOAD MB, PB (Number of intrusion attempts during that period =49.  
Consideration of vulnerability in the system.)

<Approach 1>

EFSM, Backtracking

```

                                <Result>
<CONFIDENCE listing vulnerability in the system >
                                <Low> NO</Low>
                                <Medium>NO </Medium>
                                <High> YES </High>
</CONFIDENCE>
                                </Result>
                                </Approach 1>
                                <Approach 2>
Representation in MES (vulnerability in the system
exploited)
                                <Result>
<ATTACK PROGRESSION>
<ServerVulnerability> YES Exploited </ServerVulnerability>
<Time> 10 Seconds</Time>
</ATTACK PROGRESSION>
                                </Result>
                                </Approach 2>
                                <Approach3>
                                Vulnerability tree reasoning
                                </Possible>
                                New vulnerability in the system NO
                                </Possible>
                                </Possible>
                                Existing vulnerability in the system YES
                                </Possible>
                                </Difficult> NO</Difficult>
                                <Possiable>YES<Possiable>
                                <Result>
<CONFIDENCE>
                                <Low> NO</Low>
                                <Medium>NO</Medium>

```



<High>YES</High>

</CONFIDENCE>

</Result>

</Approach3>

<Approach4>

Server Vulnerability present and exploited----- Nodes (AND, OR) ----Log File  
Contains special characters; Size is large compared to normal operation operation.

</Approach4>

<WEIGHVALUE> “9” </WEIGHTVALUE>

</COUNTERFACTUAL REASONING>

</WEIGHT VALUE = OB-1-cc >

Using the above conditions we assign and derive the weight values.

$OB - 1 - cc$ $\sum_{i=1}^n W_i$	$OB - 1 - t$ $\sum_{i=1}^n W_i$	EVENTS
7	6	Number of deleted files from log files directory from time of incident to time of analysis=45. Consideration of vulnerability in the system.
8	5	Number of file system signatures to determine if the file system was wiped=1.Consideration of antivirus software running during the incident.
7	1	Number of backdoor entry programs= 2. Consideration of vulnerability in the system.
8	5	Number of corrupted access log files=7.Consideration of Temp files deleted just after the reported incident type was high compared to normal system operation.
9	4	Number of intrusion attempts during that period =49. Consideration of vulnerability in the system.
39	21	Final Value

LEVEL 2:

Measurement Based Information (MB):

- Number of security tools installed/uninstalled=25.
- Number of event log files deleted =26

Perception Based Information (PB):

- Consideration of tools used for detection.
- Consideration of command runs remotely.
- Consideration of vulnerability in the system like double dot attacks.

Using the above conditions we assign and derive the weight values.

$OB - 1$ $\sum_{i=1}^n W_i$	EVENTS
5	Number of security tools installed/uninstalled=25.Consideration of vulnerability in the system leading to like double dot attacks
5	Number of event log files deleted =26. Consideration of command runs remotely.
10	Final Value

LEVEL 1:

Measurement Based Information (MB):

- Number of times user rights were changed=5.
- The registry file entries changed =10.

Perception Based Information (PB):

- Consideration of tools used for detection.
- Consideration of privileges assigned to the directories in the system.

Using the above conditions we assign and derive the weight values.

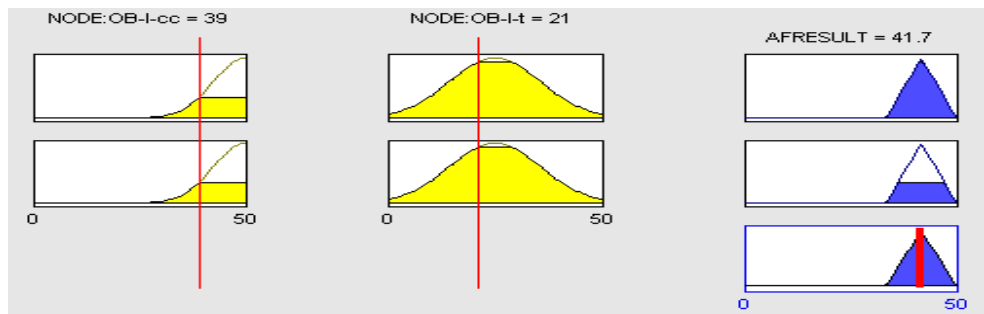
OB - 1	EVENTS
--------	--------

$\sum_{i=1}^n W_i$	
5	Number of times user rights were changed=5. Consideration of privileges assigned to the directories in the system.
6	Number of times registry entries changed=10. Consideration of tools used for deletion.
11	Final Value

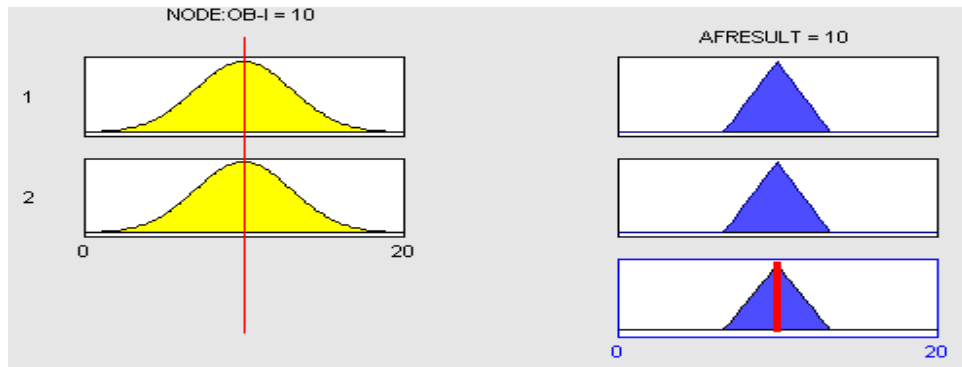
In our case the maximum value a node can take is 50 for level 3, 20 for level 2 and level 1. In our opinion we then classify low as a node value between 0 and 16, medium as a node value between 16 and 32, and high as a node value between 32 and 50 for level 3. For level 2 and level 1 we classify low as a node value between 0 and 6, medium as a node value between 6 and 12, and high as a node value between 12 and 20. The results are shown in Figure 28 above.

Using the fuzzy concept discussed in section 5.4 we arrive at the following results.

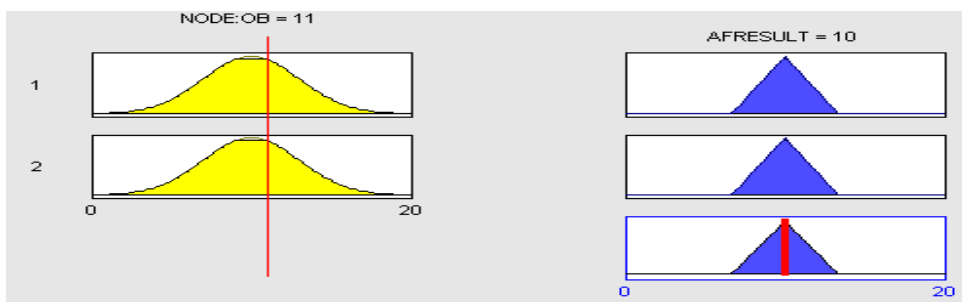
Level 3 result



Level 2 result



### Level 1 result



The above results clearly show the likeliness of AF activity in the system and the confidence in the evidence collected is invalid. The investigators reason was that the last two usernames present in the access log files are user Y's so he has edited the system.

Using our meta-forensic approach and the likeliness of AF is high and analysing the logic shows the presence of system vulnerability which was successfully exploited. So the conclusion that Y edited the system is not true. This approach provided the forensic investigator to look for more information on vulnerability exploited and the claim Y edited the system is not acceptable.

## 5.9 System Automation

Since the process of validation described in above sections takes multiple steps to complete and becomes difficult for large scale systems we propose a plan for automation of the meta-forensic model. The anti-forensic model known as validator can be architected to run in online or offline mode. The framework of the architecture is derived from a combination of security and forensic models which were reviewed in literature survey. Since all the models have a common framework in place we arrive at following high level architecture.

From literature in section 3.10 and 3.15 the policy based models and the CIM user and security model provides a set of relationships between various users, their credentials, the managed elements that represent the resources, and the resource managers involved. The validator is shown in Figure 32 below. The collected evidence follows the standard investigation procedure and is fed into a database after classification. Most of the classifications already exist and defined in CIM model and the new classification can easily be added. A web based console can be used to interact with database to group MB, PB value and assign weights. The values can then be run as a fuzzy operation to obtain results. This high level architecture can be implemented using any software program.

The investigated system which is to be validated is connected to the forensic process analyser. This can be online or offline mode. On online mode this can be connected to a live network feed. In offline mode it can be a standalone laptop, computer or server. A set of MB and PB values are set. Once the value is set the weight is assigned logically. The algorithm discussed in section 5.4

is used to set weights. This can be programmed in a high level language for system automation. After the completion of weight assignment the logical operations are carried out in fuzzy tool box to produce the results.

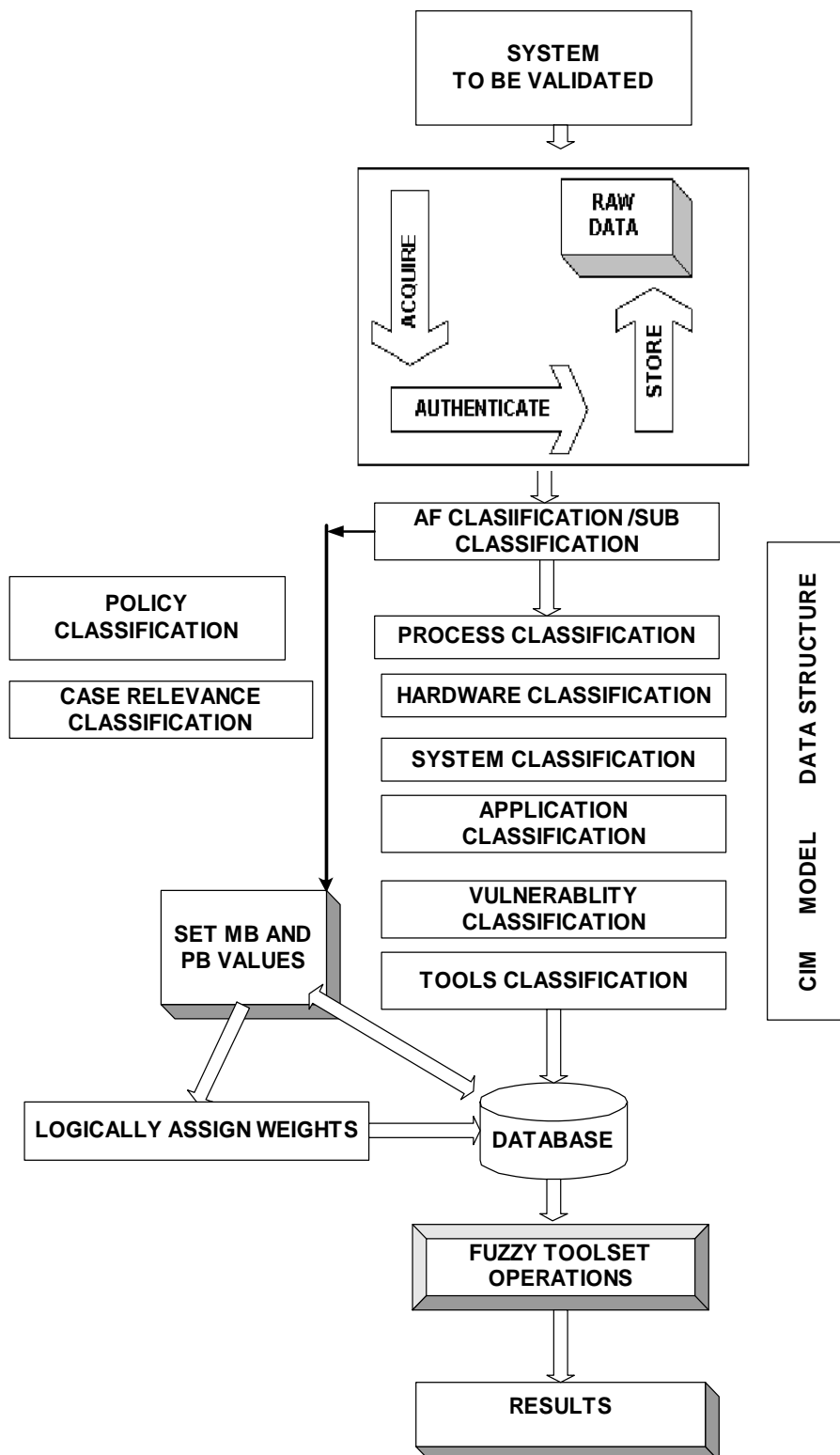


Figure 32: System Automation of Meta-Forensic Validity Model

## **5.10 Summary**

The meta-forensic model was presented in this chapter. An overview of classifications was briefly given, before an account of the decision tree and level of depth requirements. The existing classification of the AF was extended to sub categories and a new language to denote AF was established. The various ways of obtaining information were also examined, both perception based and other measurement techniques. The process of assigning weights was the next important step in the model. A reasoning based approach was proposed to assign weights. It is crucial that this stage is accurately executed in order for nodes to have the right values. Fuzzy logic was applied. Following which the model application was presented with some of the AF cases. The meta-forensic process was proposed as high level architecture as an automated system which can work in online or offline mode. The system automation of the meta-forensic model can work alongside with IDS and IPS system to detect active AF activity in network.



## **CHAPTER 6: CONCLUSION AND FUTURE WORK**

Validation models examine and validate the evidence and use their relationships in order to construct states and dependencies that will not only validate evidence but also gives us complete scenarios of a threat or exploited loophole in security. Common vulnerabilities in the states can be identified during the process and countered in order to secure the system in a cost effective manner after the investigation. The current models and procedures lack well-structured forensic validation capabilities, and this research has addressed those issues with a new approach to meta-forensic approach. The proposed meta-forensic model is intended to augment the traditional forensic steps of Acquire, Authenticate, and Analyse. It has been argued that digital forensic evidence integrity is compromised by the threat of anti-forensic attacks.

The proposed new methods of meta-forensics are required to address the validity of evidence collected. The outcome of this research contributes to the existing knowledge of forensics anti-forensics and security as a whole. As little published work is available in the meta-forensics industry this research defines the concept of anti-forensic approach in new perspective. The process of validation helps the investigators to look at undetected AF in a given system during any forensic investigation.

Chapter 2 and 3 shows us the current knowledge in this field. Chapter 4 defines the problem of why we require a validation and how the security of a system can be compromised when a

systematic attack is deployed. VMware simulated the hardware specifications for the experiment without huge financial costs. The result of the experiment gives an insight into how a hacker can cause disruption to a security system and avoid detection. The experiment also demonstrates the capabilities of various tools, measures and countermeasures available in security field for forensic and anti-forensic activity. The reliability of the current forensic and anti forensic tools is questioned. The experiment demonstrated the concept of hack and system compromise and systematic failures to detect AF activity. The threat of AF which was proposed in hypothesis was demonstrated as very much real in this chapter. With the problem domain defined we set out to propose a solution in chapter 5.

Currently there are no tools or technology in the market to address this issue and the AF model will be a great application. The future work will involve integrating this approach to forensic software, including Encase. A commercial tool can also be developed using the proposed concept. Since the concept is defined it can be extended to other areas like mobile phone, cameras, and PDA's. A working model derived from this idea could become one of the most sought after tool in forensic security. Conclusively, this thesis succeeded in providing guidance on how to develop and implement an effective validation of digital evidence using the meta-forensic model. The meta-forensic model once approved by UK Judicial system can be used a standard to detect anti-forensics which can then be admitted in court of law to demonstrate the confidence in evidence gathered.

The meta-forensic model used in the experiment was divided up into sub-groups and components. The components consisted of classification of AF as categories and sub categories, weight assignment using logical reasoning and a fuzzy operation to obtain results. A crucial stage is setting the MB and PB values. Weights are assigned logically. A logical method

proposed by setting MB, PB values and weights allow a structured description of the investigative process is advancement to Digital Investigation Process Language. The added functionality to classify the forensic activity by looking into vulnerabilities and exploits gives a complete picture of the system being investigated. Looking into vulnerabilities and sub components of it helps us to formulate a better system security for future systems. The entire meta-forensic concept can be implemented using any software programs. The high level architecture is proposed at the end of Chapter 5.

From our research it has been clearly identified that systems can be compromised networks can be spoofed. As the responsibility lies with users of the system meta-forensic principals can be applied to strengthen computer security. As the model is portable it can be extended to number of systems irrespective of operating system, or hardware used.

Finally contribution of this thesis for the field of research is summarized.

- Currently investigators had no formal and systematic way to detect AF. Forensic and security models lack AF capabilities and this research has advanced the concept by developing meta-forensic model.
- This research contributes to a new formal language to represent AF categories and its sub categories.
- A new method, combination method of testing tools has emerged from this research. This demonstrates the confidence in tools used during forensic investigation and implementation of security.
- This thesis has advanced the concept of security in network by proposing system automation to detect active AF using meta-forensic approach.

Nevertheless, considering the relatively unexplored nature of the field, and the large gaps in the literature, the meta-forensic model presented here has gone some way in turning detection, as Sir Arthur Conan Doyle says, into a science.

## References

- [1] H M Wolfe, “Web Solutions and Technologies After the Hack”, presented at *ICE Conference*, 2003. Available: [http://www.iceconference.com/Speaker\\_Presentations/Solutions\\_Technologies/Wed\\_Solutions\\_and\\_Technologies\\_After\\_The\\_Hack\\_Forensics.ppt](http://www.iceconference.com/Speaker_Presentations/Solutions_Technologies/Wed_Solutions_and_Technologies_After_The_Hack_Forensics.ppt) [Accessed: May 25, 2010]
- [2] Thomas Alfred Johnson. *Forensic Computer Crime Investigation*. Connecticut: CRC Press 2006, p. 57.
- [3] Ryan Harris. “Arriving at an Anti-forensics consensus”, presented at *DFRWS Conference*, 2006, Available: <http://www.dfrws.org/2006/proceedings/6-Harris-pres.pdf> [Accessed: May 25, 2010] G. Palmer . “A Road Map for Digital Forensics Research”. *Digital Forensic Research Workshop (DFRWS) Technical Report (DTR) T001-01 Final*, 2001. Available: <http://www.dfrws.org/2001/dfrws-rm-final.pdf> [Accessed: September 3, 2008]
- [4] <http://www.homeoffice.gov.uk/counter-terrorism/current-threat-level/> [Accessed: APRIL, 26 2011]
- [5] Kelly J. Kuchta, “Computer Forensics Today”. *Information System Security* [serial online], 9(1):29, 2000.
- [6] Albert J. Marcella, , & Robert S. Greenfield, *Cyber forensics: A Field Manual for Collecting, Examining, and Preserving Evidence of Computer Crimes*. Boca Raton, FL: Auerbach Publications, 2002, p.169.
- [7] Ryan Harris – DFRWS 2006 conference proceedings, Arriving at an Anti-forensics consensus, available at: <http://www.dfrws.org/2006/proceedings/6-Harris-pres.pdf> [Accessed: May, 20 2009]
- [8] Kuchta, Kelly J. Computer Forensics Today .*Information System Security* [serial online], 9 (1):29, 2000.
- [9] Marcella, Albert J., & Greenfield, Robert S. CYBER FORENSICS: A Field Manual for Collecting, Examining, and Preserving Evidence of Computer Crimes. Boca Raton, FL: Auerbach Publications, 2002 Page 169.
- [10] Vacca, John R. *Computer Forensics: Computer Crime Scene Investigation*. Hingham, MA: Charles River Media, 2002. Page 4.

- [11] Caloyannides, Michael A. Privacy Protection and Computer Forensics. Boston, MA: Artech House, 2004. Page 1.
- [12] Welch, T. Computer Crime Investigation and Computer Forensics. Information Systems Security [serial online], 6(2):56, 1997.
- [13] An Examination of Digital Forensic Models Mark Reith, Clint Carr, Gregg Gunsch International Journal of Digital Evidence Fall 2002, Volume 1, Issue 3.
- [14] Silverstone, H & Davia, H. (2005) Fraud 101, 2nd Ed. Hoboken: John Wiley & Sons, Inc
- [15] Vacca, J. (2005) Computer Forensics: Computer Crime Scene Investigation, 2nd Ed. Hingham: Charles River Media.
- [16] Weil, M.C. Dynamic time & date stamp analysis. International Journal of Digital evidence 1,2(2002) <http://www.ijde.org> Last accessed May 20<sup>th</sup> 2009.
- [17] [http://en.wikipedia.org/wiki/Burden\\_of\\_proof#Standard\\_of\\_proof](http://en.wikipedia.org/wiki/Burden_of_proof#Standard_of_proof). Last accessed May 20<sup>th</sup> 2009
- [18] <http://www.parliament.uk/documents/post/postpn271.pdf>. Last accessed June 20<sup>th</sup> 2011
- [19] All in One CISSP Shon Haris Fifth Edition Publishing 2009.
- [20] "ACPO Good Practice Guide for Computer-Based Evidence". ACPO. [http://www.7safe.com/electronic\\_evidence/ACPO\\_guidelines\\_computer\\_evidence\\_v4\\_web.pdf](http://www.7safe.com/electronic_evidence/ACPO_guidelines_computer_evidence_v4_web.pdf). Last accessed June 16<sup>th</sup> 2011
- [21] <http://tools.ietf.org/html/rfc1094>. Last accessed June 20<sup>th</sup> 2011
- [22] "Double Encoding". Available: [http://www.owasp.org/index.php/Double\\_Encoding](http://www.owasp.org/index.php/Double_Encoding). [Accessed: May, 25, 2010]
- [23] Liu, V., & Brown, F. (2006, April 3). Bleeding-Edge Anti-Forensics. Presentation at InfoSecWorld 2006. Available: [www.stachliu.com/files/InfoSecWorld\\_2006-K2-Bleeding\\_Edge\\_AntiForensics.ppt](http://www.stachliu.com/files/InfoSecWorld_2006-K2-Bleeding_Edge_AntiForensics.ppt). [Accessed: February, 20 2010]
- [24] Rogers, M. (2006, March 22). Panel session at CERIAS 2006 Information Security Symposium. <http://www.cerias.purdue.edu/symposium/2006/materials/pdfs/antiforensics.pdf> [Accessed: February, 22 2010]
- [25] [http://findarticles.com/p/articles/mi\\_qa3926/is\\_199810/ai\\_n8820687/](http://findarticles.com/p/articles/mi_qa3926/is_199810/ai_n8820687/) Last accessed Feb 20<sup>th</sup> 2010

- [26] Kessler, G.C. (2004, July). An Overview of Steganography for the Computer Forensics Examiner. *ForensicsScienceCommunication*,6(3).  
[http://www.fbi.gov/hq/lab/fsc/backissu/july2004/research/2004\\_03\\_research01.htm](http://www.fbi.gov/hq/lab/fsc/backissu/july2004/research/2004_03_research01.htm) [Accessed: February, 20 2010]
- [27] Rogers, D. M. (2005). Anti-Forensic Presentation given to Lockheed Martin. San Diego.  
[http://www.cyberforensics.purdue.edu/documents/AntiForensics\\_LockheedMartin09152005.pdf](http://www.cyberforensics.purdue.edu/documents/AntiForensics_LockheedMartin09152005.pdf)  
[Accessed: February, 20 2010]
- [28] <https://www.dss.mil/GW/ShowBinary/DSS/isp/odaa/documents/nispom2006-5220.pdf#page=75> [Accessed: February, 20 2010]
- [29] <http://sourceforge.net/> [Accessed: February, 22 2010]
- [30] Kissel, R., Scholl, M., Skolochenko, S., & Li, X. (2006). Guidelines for Media Sanitization. Gaithersburg: Computer Security Division, National Institute of Standards and Technology.
- [31] “Anti-computer Forensics”. Wikipedia. Available: [http://en.wikipedia.org/wiki/Anti-computer\\_forensics](http://en.wikipedia.org/wiki/Anti-computer_forensics) [Accessed: May, 25, 2010]
- [32] Berinato, S. (2007). The Rise of Anti Forensics.  
[http://www.csoonline.com/article/221208/The\\_Rise\\_of\\_Anti\\_Forensics](http://www.csoonline.com/article/221208/The_Rise_of_Anti_Forensics). [Accessed: February, 20 2010]
- [33] Guidance Software. (2007, July 26). Guidance Software Response to iSEC Report.  
<http://www.securityfocus.com/archive/1/474727> Last accessed Feb 22th 2010
- [34] Palmer, C., Newsham, T., Stamos, A., & Ridder, C. (2007, August 1). Breaking Forensics Software: Weaknesses in Critical Evidence Collection. Abstract of presentation at Black Hat USA 2007.  
<http://www.blackhat.com/html/bh-usa-07/bh-usa-07-speakers.html#Palmer> [Accessed: February, 22 2010]
- [35] <http://www.metasploit.com/framework/> Last accessed Feb 22th 2010
- [36] Information Security Governance. Guide for Board of Directors.  
<http://www.isaca.org/Knowledge-Center/Research/Documents/InfoSecGuidanceDirectorsExecMgt.pdf>  
[Accessed: July, 28 2010]
- [37] Zadeh, L.A. Fuzzy Sets and Applications: Selected Papers of L.A. Zadeh, ed. by Yager, Ovchinnikov, R.M. Tong, H.T. Nguyen, John Wiley and Sons, 1987.

[38] Zadeh Lotfi, "Coping with the Imprecision of the Real World: An Interview with Lotfi A. Zadeh" *Communications of the ACM* (April 1984).

[39] Negoita, Constantin. *Expert Systems and Fuzzy systems* Benjamin/Cummings Publishing Company, 1985.

[40] Kandel, "Fuzzy Statistics and Policy Analysis", *Fuzzy Sets: Theory and Applications to Policy Analysis and Information Systems*, ed. by P. Wang and S. Chang, Plenum Press, New York. 1980.

[41] Ronald L. Krutz, and Russell Dean Vines, , *The CISSP Prep Guide; Gold Edition*. Indianapolis: Wiley Publishing 2003.

[42] A. Kleppe, W. Bast, J. B. Warmer, and A. Watson. *MDA Explained: The Model Driven Architecture—Practice and Promise*. Addison-Wesley, 2003.

[43] D. A. Basin, J. Doser, and T. Lodderstedt. Model driven security: From UML models to access control infrastructures. *ACM Trans. Softw. Eng. Methodol.*, 15(1):39–91, January 2006.

[44] G. Georg, I. Ray, and R. France. Using aspects to design a secure system. In *ICECCS '02: Proceedings of the Eighth International Conference on Engineering of Complex Computer Systems*, pages 117–126, Washington, DC, USA, 2002. IEEE Computer Society.

[45] J. Jurjens. Towards development of secure systems using UMLsec. In *Fundamental Approaches to Software Engineering (FASE/ETAPS 2001)*, volume 2029 of LNCS, pages 187–200. Springer, 2001.

[46] J. Jurjens. UMLsec: Extending UML for secure systems development. In *UML 2002 — The Unified Modeling Language*, volume 2460 of LNCS, pages 412–425. Springer, 2002.

[47] D. A. Basin, J. Doser, T. Lodderstedt, Model driven security: From UML models to access control infrastructures., *ACM Transactions on Software Engineering and Methodology* 15 (1) (2006) 39–91

[48] M, Noblett, Pollitt, M, Presley, "Recovering and Examining Computer Forensic Evidence", *Forensic Science Communications*, Volume 2 Number 4, 2000.



- [49] Digital Forensic Research Workshop (DFRWS) Utica: Research Road Map, 2001
- [50] Stephenson, P. “Modeling of Post-Incident Root Cause Analysis”, *International Journal of Digital Evidence*, Volume 2, Issue 2,m 2003
- [51] “What is design? CPN” Available: <http://www.daimi.au.dk/designCPN/what.html>. [Accessed: February, 22 2010]
- [52] Carrier, B. and Spafford, E. “An Event-based Digital Forensic Investigation Framework”, DFRWS, Baltimore, MD Center for Education and Research in Information Assurance and Security - CERIAS,Purdue University West Lafayette, 2004
- [53] Ruibin, G., Yun C., and Gaertner, M. “Case-Relevance Information Investigation: Binding Computer Intelligence to the Current Computer Forensic Framework”, *International Journal of Digital Evidence Spring*, Volume 4, Issue 1, 2005
- [54] Bishop M. “Introduction to computer security”. Addison-Wesley; 2004.
- [55] D. Elliott Bell and Leonard J. LaPadula “Secure Computer Systems: Mathematical Foundations”. MITRE Technical Report 2547, Volume I, 1973.
- [56] Biba, K.J. "Integrity Considerations for Secure Computer Systems", MTR-3153, The Mitre Corporation, April 1977
- [57] Lipton, Richard J.; Snyder, Lawrence. “A Linear Time Algorithm for Deciding Subject Security” (PDF). *Journal of the ACM* 24 (3): 455-464. Addison-Wesley, 1977
- [58] Dorothy E Denning. An Intrusion Detection Model. In *IEEE Transactions on Software Engineering*, Number 2, page 222, February 1987.
- [59] Wei Ren Hai Jin “Modeling the Network Forensics Behaviours” Department of Computer Science, Zhongnan Univ. of Economics & Law, P.R. China 2School of Computer Science, Huazhong Univ. of Science & Technology, P.R. China
- [60] Wei Ren Hai Jin “Modelling the Network Forensics Behaviours” IEEE publication, 2005
- [61] “DMTF Standards and Initiatives” Available: <http://www.dmtf.org/standards> [Accessed: February, 22 2010].

- [62] S Vidalis and A Jones. "Using Vulnerability Trees for Decision Making in Threat Assessment" School of Computing Technical Report CS-03-2
- [63] Computer Programming Software Terms, Glossary and Dictionary- EFSM: Extended Finite State Machine Model". Available:  
<http://www.networkdictionary.com/software/e.php?PHPSESSID=9926acc9b2e4f8f05ced05a620abcfe9>. [Accessed: February, 22 2010].
- [64] Gladyshev P., Patel A. "Finite state machine approach to digital event reconstruction". Digital Investigation Journal 2004; 1(2)
- [65] Counterfactual Theories of Causation Hume D (originally published 1748, Section VII) First published Wed Jan 10, 2001. Available:  
<http://plato.stanford.edu/entries/causation-counterfactual/#EarCouThe> [Accessed: February, 22 2010]
- [66] Thilo Paul-Stüve. Formal Task Analysis of Graphical System Engineering Software Use. Rapport technique, RVS Group, Faculty of Technology, University of Bielefeld, March 2005. Available:  
<http://www.rvs.uni-bielefeld.de/publications/books/WBAbook/> [Accessed: February, 22 2010].
- [67] Schneier, Bruce (December 1999). Attack Trees. Dr Dobb's Journal, v.24, n.12..  
<http://www.schneier.com/paper-attacktrees-fig2.html> [Accessed: February, 22 2010]
- [68] <http://www.rvs.uni-bielefeld.de/research/WBA/> [Accessed: February, 22 2010]
- [69] <http://www.universalteacherpublications.com/univ/free-assign/cs70/page3.htm> [Accessed: August 3, 2010]
- [70] Models For Accident Investigation Prepared by Michael D. Harvey, Ph.D. Under contract to the Research Branch April, 1985.
- [71] NIST Computer Forensics Tool Testing Project. <http://www.cftt.nist.gov>. Last accessed Jan 2, 2010.
- [72] Brian Carrier. Digital forensics tool testing images. <http://dfft.sourceforge.net/>, [Accessed: December, 22 2009].
- [73] <http://www.porcupine.org/forensics/tct.html> [Accessed: February, 27 2010].
- [74] <https://www.eeye.com> [Accessed: December, 22 2009].
- [75] SnortTM The Open Source Network Intrusion Detection System  
<http://www.snort.org> [Accessed: December, 22 2009].

- [76] Marcus K Rodgers – Lockheed presentation, Anti-forensics, available at: [www.cyberforensics.purdue.edu](http://www.cyberforensics.purdue.edu) [Accessed: December, 22 2009].
- [77] <http://www.cloudantivirus.com/en/listofviruses/> [Accessed: December, 22 2009].
- [78] [http://www.symantec.com/business/security\\_response/threatexplorer/threats.jsp](http://www.symantec.com/business/security_response/threatexplorer/threats.jsp) Last accessed Dec 22, 2009.
- [79] <http://www.wildlist.org/WildList/> [Accessed: December, 22 2009].
- [80] Coad, P. and E. Yourdon (1991). Object-Oriented Analysis, Prentice Hall Inc.
- [81] Zadeh, L.A., 2005. Toward a generalized theory of uncertainty - an outline. Inf. Sciences 172, 1–40.
- [82] TAMMPS: a threat assessment model for micro-payment systems. School of Computing. Pontypridd, University of Glamorgan: 1-152, Vidalis, S. (2001).
- [83] L. A. Zadeh, “Fuzzy logic computing with words,” IEEE Trans. Fuzzy Syst., vol. 4, no. 2, pp. 103–111, 1996.
- [84] Zadeh, L.A, Calculus of fuzzy restrictions, in: Fuzzy Sets and Their Applications to Cognitive and Decisions Process, edited by L.A. Zadeh, K.S. Fu, K. Tanaka and M. Shimura, Academic Press, New York, 1975.
- [85] Zadeh, L.A, Fuzzy sets, Information and Control 8(1965), 338-353.
- [86] Zadeh, L.A., 2002. Toward a perception-based theory of probabilistic reasoning with imprecise probabilities. J. Statist. Plann. Inference 105, 233–264.
- [87] “What is a virtual machine?”. VMware. Available: <http://www.vmware.com/virtualization/virtual-machine.html> [Accessed: May, 25, 2010]

## **Appendix A**

### **Network Topology**

Possible network topologies for the device set up are:

#### **Bus**

The simplest topology to understand is the Bus. In a Bus, all the devices on the network are connected to a common cable. Normally, this cable is terminated at either end, and can never be allowed to form a closed loop.

#### **Ring**

A Ring topology is very similar to the Bus. In a Ring, all the devices on the network are connected to a common cable which loops from machine to machine. After the last machine on the network, the cable then returns to the first device to form a closed loop.

#### **Star**

A star topology is completely different from either a Bus or a Ring. In a star each device has its own cable run connecting the device to a common hub or concentrator. Only one device is permitted to use each port on the hub.

#### **Tree**

A tree topology can be thought of as being a "Star of Stars" network. In a Tree network, each device is connected to its own port on a concentrator in the same manner as in a Star. However,

concentrators are connected together in a hierarchical manner, i.e., a hub will connect to a port on another hub.

### **Mesh**

A Mesh topology consists of a network where every device on the network is physically connected to every other device on the network. This provides a great deal of performance and reliability, however, the complexity and difficulty of creating one increases geometrically as the number of nodes on the network increases. For example, a three or four node mesh network is relatively easy to create; whereas it is impractical to set up a mesh network of 100's of nodes, as the number of interconnections would be so ungainly and expensive.

## Appendix B

### VMware

In Chapter 4, the cost of the experiment was described in connection with VMware. VMware was used as a more financially viable alternative **than building** a physical laboratory with actual hardware. What VMware software allows for is a completely **virtualised** set of hardware to the guest operating system. **In this way the experiment could be conducting with the necessary hardware specifications without the initial overhead costs of actually buying and installing the hardware** [87]. Virtualisation was first developed in the 1960s to partition large, mainframe hardware for better hardware utilisation [87]. VMware invented virtualisation in the 1990s to address underutilization and other issues, overcoming many challenges in the process. VMware software virtualises the hardware for a video adapter, a network adapter, and hard disk adapters. The host provides pass-through drivers for guest USB, serial, and parallel devices. In this way, VMware virtual machines become highly portable between computers, because every host looks nearly identical to the guest. In practice, a system administrator can pause operations on a virtual machine guest, move or copy that guest to another physical computer, and there resume execution exactly at the point of suspension.

So in the experiment the virtual machines used were running operating systems and applications as if it they a physical computer. A virtual machine behaves exactly like a physical computer and contains it own virtual CPU, RAM and hard disk [87]. It was perfect for this AF study as operating systems can't tell the difference between a virtual machine and a physical machine, nor can applications or other computers on a network. Even the virtual machine thinks it is a "real"

computer [87]. Nevertheless, a virtual machine is composed entirely of software and contains no hardware components whatsoever. As a result, VMware was used in this experiment has entire feature for a digital anti-forensic experiment were present without the prohibitive capital expense.

## Appendix C

### WINDOWS 2000 SCAN REPORT

# Retina - Network Security Scanner

*Network Vulnerability Assessment & Remediation Management*

06/03/2010 - Report created by Retina version 5.11.3.2195

Metrics for 'FULL SCAN'	
File name:	D:\Program Files\eye Digital Security\Retina 5\Scans\WIN2000.rtd
Audits revision:	2195
Scanner version:	5.11.3
Start time:	06/03/2010 23:52:55
Duration:	0d 0h 1m 52s
Credentials:	6B3459551CAD439098C4B8AD6977FE10
Audit groups:	All Audits
Address groups:	N/A
IP ranges:	192.168.1.6
Total hosts attempted:	1
Total hosts scanned:	1
No access:	0

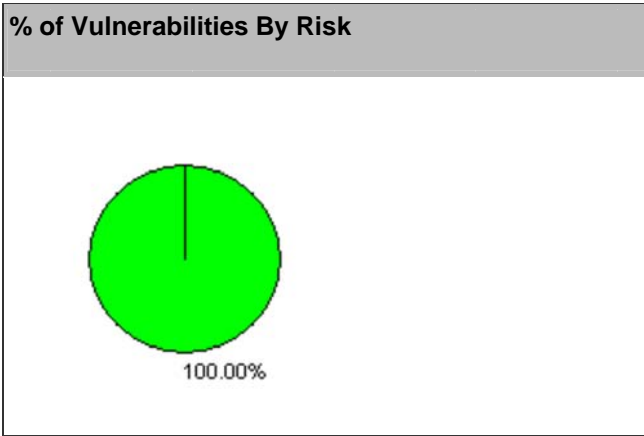
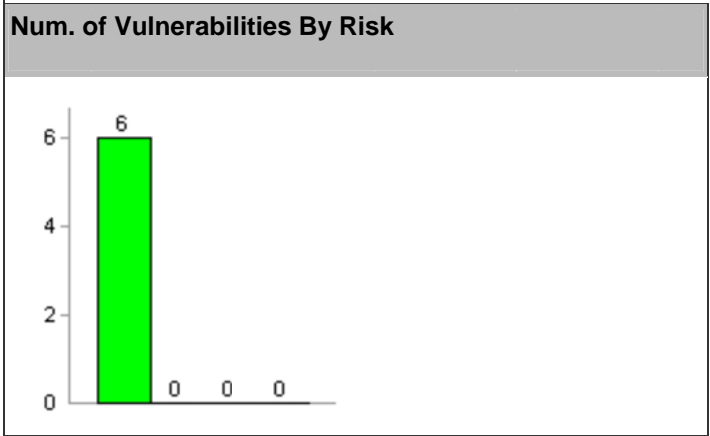
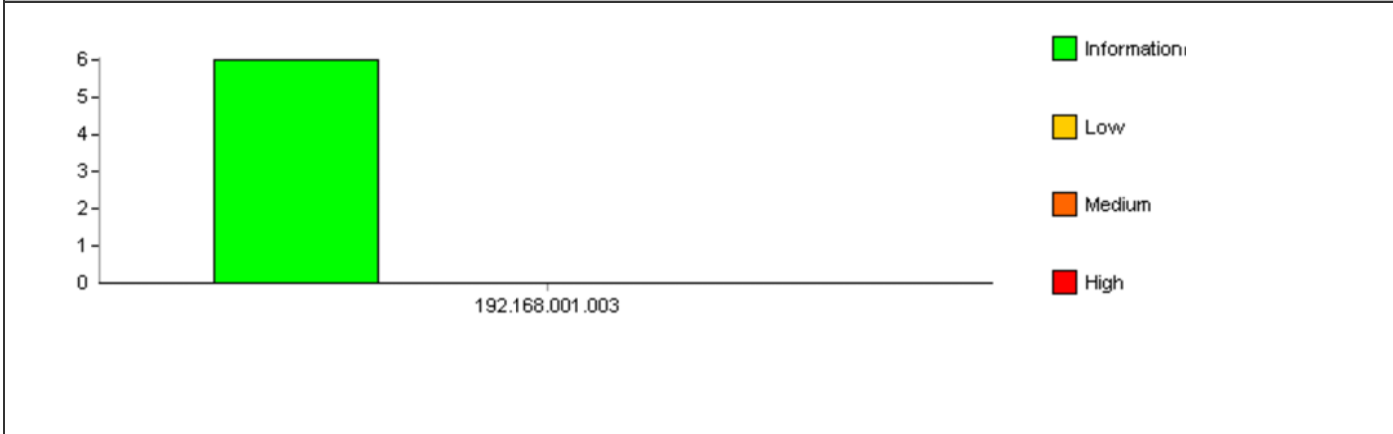


## NETWORK ANALYSIS RESULTS

Scanner Name	Retina	Machines Scanned	1
.			
Scanner Version	5.11.3.2195	Vulnerabilities Total	273
.			
Scan Start Date	06/03/2010	High Risk Vulnerabilities	152
.			
Scan Start Time	23:52:55	Medium Risk Vulnerabilities	64
.			
Scan Duration	0h 1m 52s	Low Risk Vulnerabilities	57
.			
Scan	FULL	Information	26

Name	SCAN	Only Audits	
Scan Status	Completed	Credential Used	6B3459551CAD439098C4B8AD6977FE10

### Top 5 Most Vulnerable Hosts



## RHEL LINUX SCAN REPORT

# Retina - Network Security Scanner

*Network Vulnerability Assessment & Remediation Management*

07/03/2010 - Report created by Retina version 5.11.3.2195

### Metrics for 'FULL SCAN'

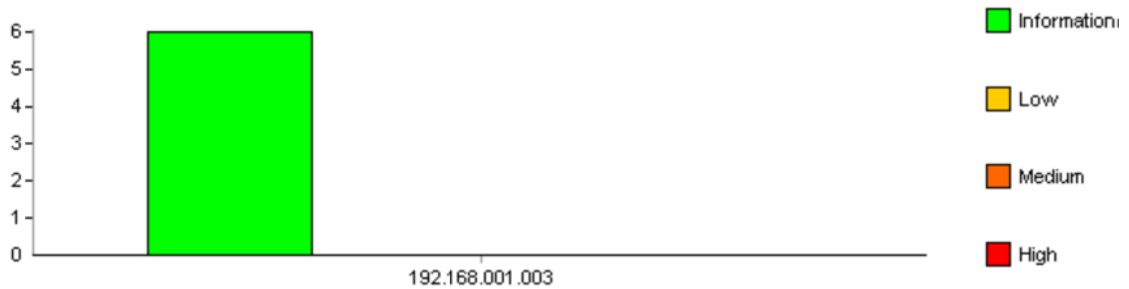
File name:	D:\Program Files\Eye Digital Security\Retina 5\Scans\linux.rtd
Audits revision:	2195
Scanner version:	5.11.3
Start time:	07/03/2010 00:41:33
Duration:	0d 0h 12m 25s
Credentials:	289C25B453FC41B7A4A2A68E778201AB
Audit groups:	All Audits
Address groups:	N/A
IP ranges:	192.168.1.7
Total hosts attempted:	1
Total hosts scanned:	1
No access:	1

## NETWORK ANALYSIS RESULTS

### Report Summary

Scanner Name	Retina	Machines Scanned	1
Scanner Version	5.11.3.2195	Vulnerabilities Total	2
Scan Start Date	07/03/2010	High Risk Vulnerabilities	0
Scan Start Time	00:41:33	Medium Risk Vulnerabilities	1
Scan Duration	0h 12m 25s	Low Risk Vulnerabilities	1
Scan Name	FULL SCAN	Information Only Audits	1
Scan Status	Completed	Credential Used	289C25B453FC41B7A4A2A68E778201AB
Vulnerable Machines 1			

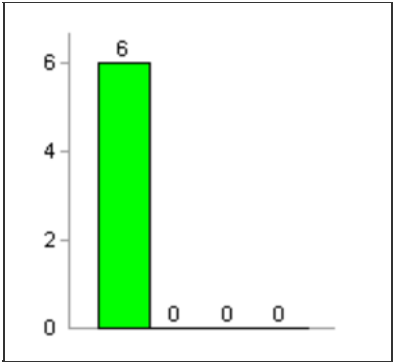
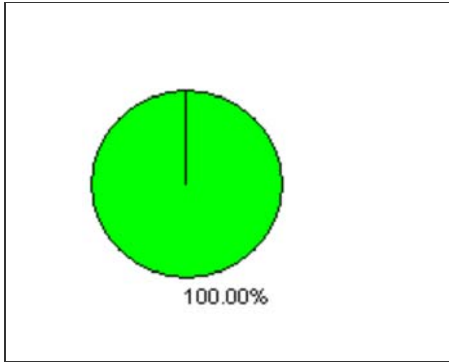
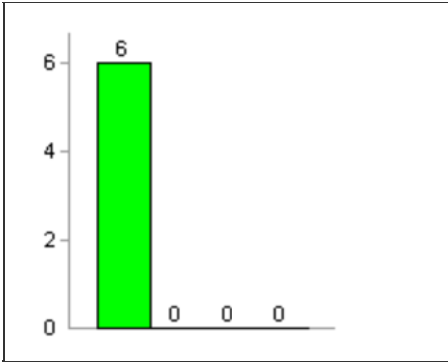
### Top 5 Most Vulnerable Hosts



Num. of Vulnerabilities By Risk

% of Vulnerabilities By Risk

Avg. of Vulnerabilities By Risk

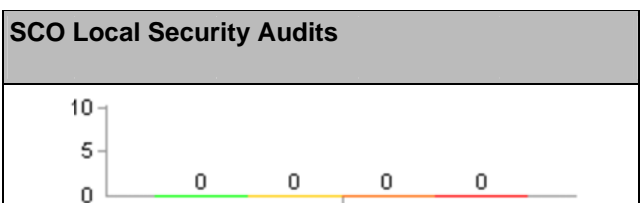
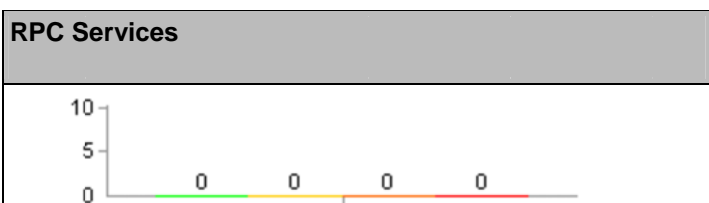
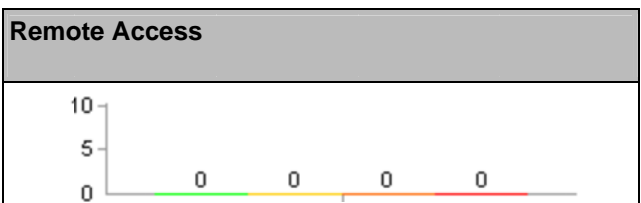
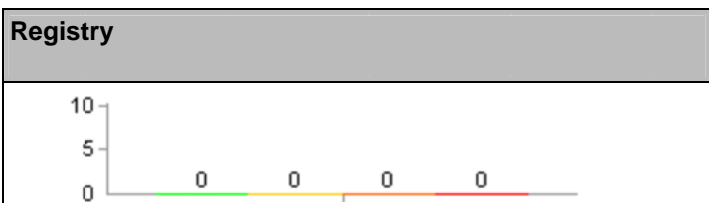
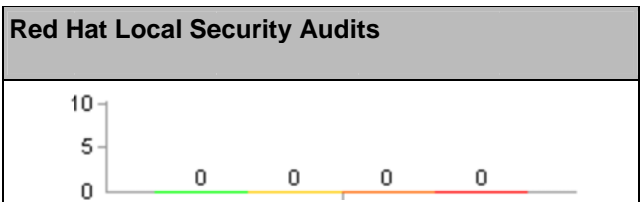
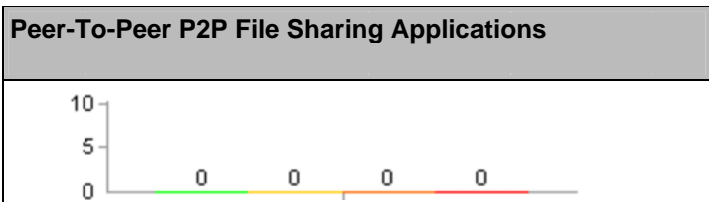
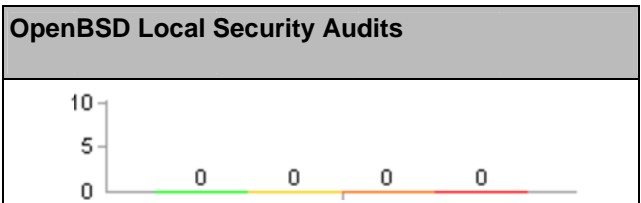
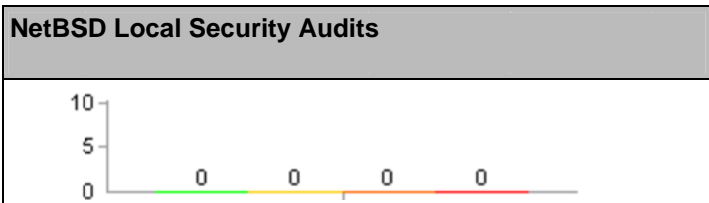
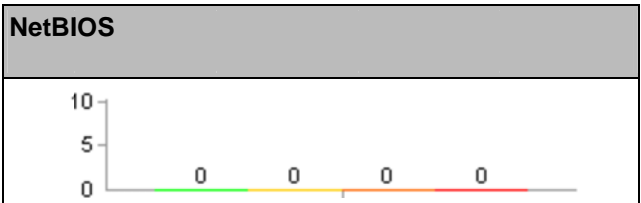
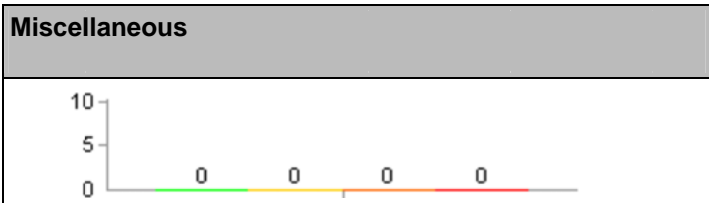
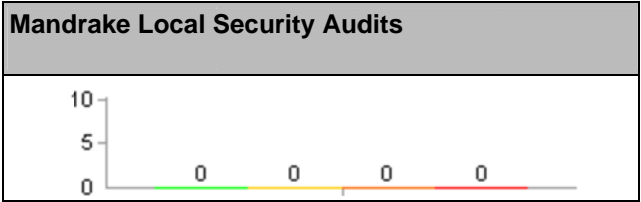
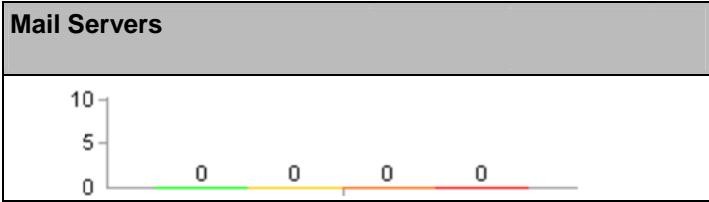
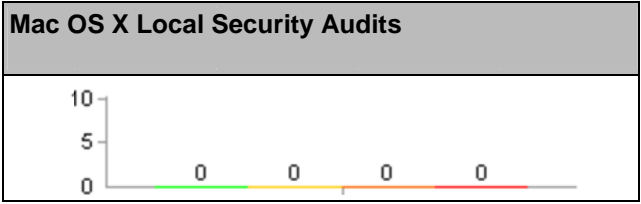
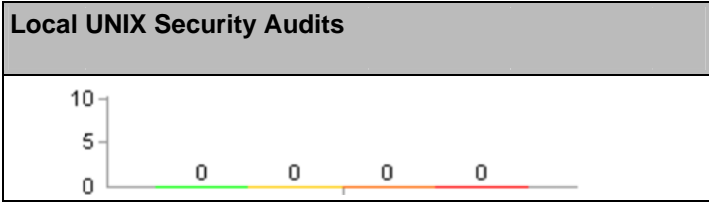


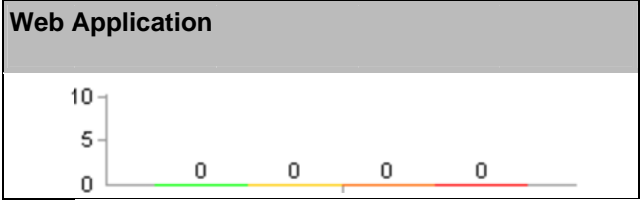
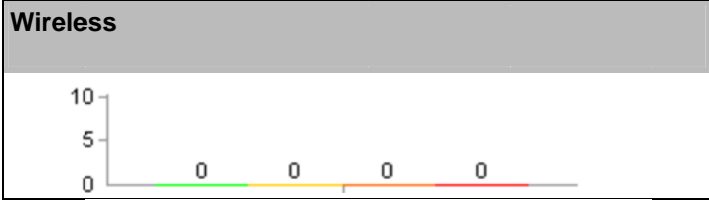
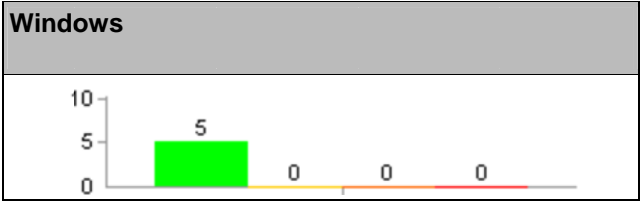
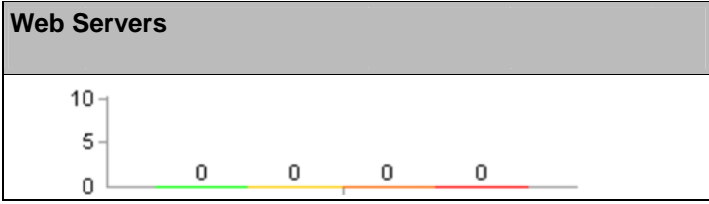
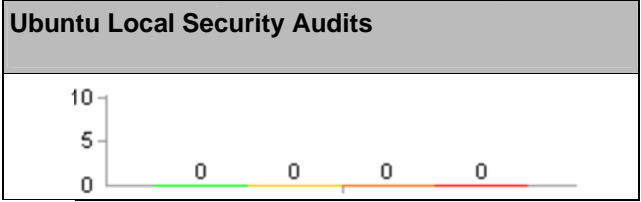
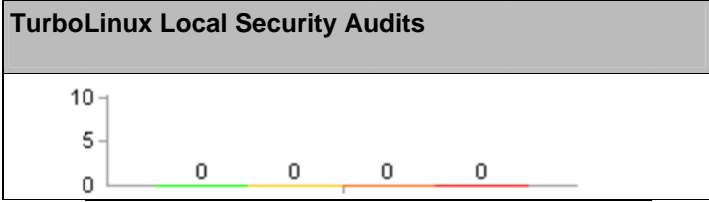
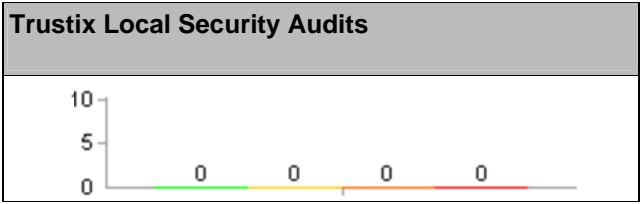
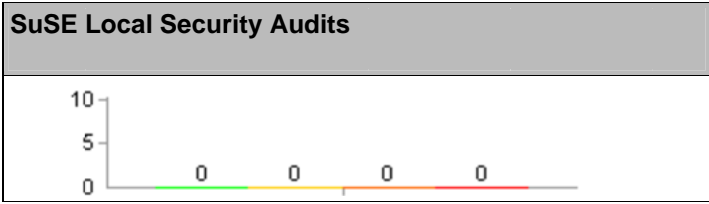
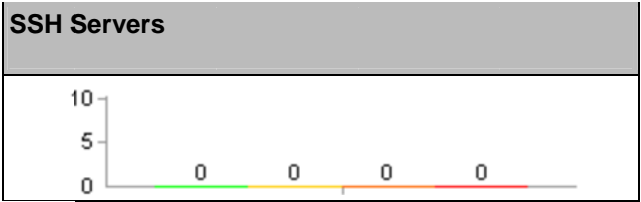
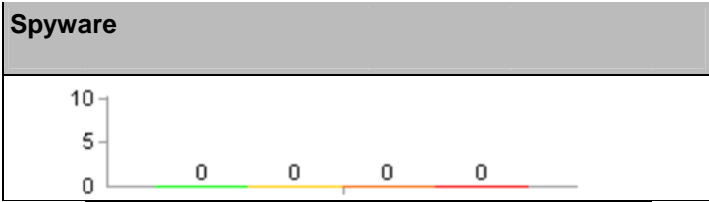
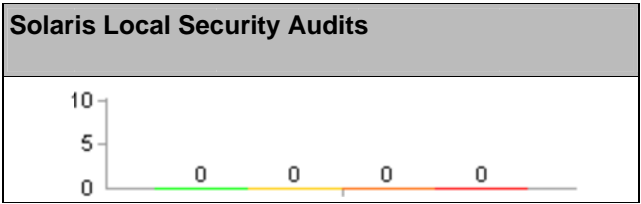
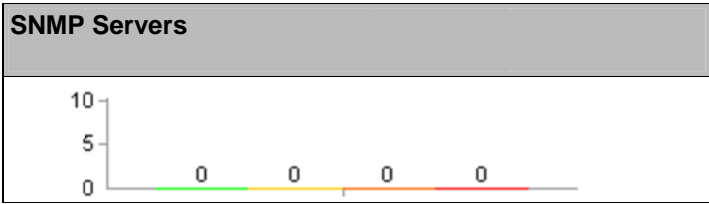
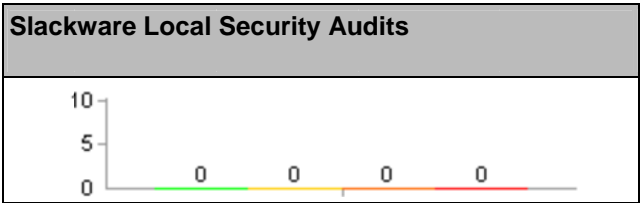
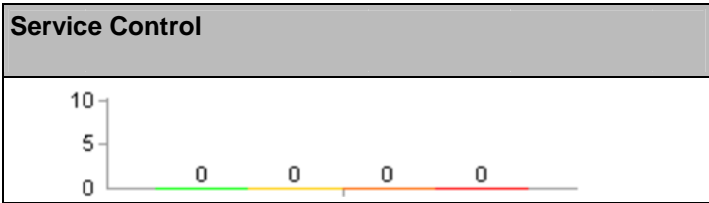
**TOTAL VULNERABILITIES BY CATEGORY**

The following is an overview of the total vulnerabilities by audit category.

<p><b>Accounts</b></p> <table border="1"> <tr><td>1</td><td>0</td><td>0</td><td>0</td></tr> </table>	1	0	0	0	<p><b>AIX Local Security Audits</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0
1	0	0	0						
0	0	0	0						
<p><b>Anti-Virus</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	<p><b>Backdoors</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0
0	0	0	0						
0	0	0	0						
<p><b>Caldera Local Security Audits</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	<p><b>CGI Scripts</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0
0	0	0	0						
0	0	0	0						
<p><b>Cisco Local Security Audits Cisco Local Security Audits</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	<p><b>Conectiva Local Security Audits</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0
0	0	0	0						
0	0	0	0						
<p><b>Database</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	<p><b>Debian Local Security Audits</b></p> <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0
0	0	0	0						
0	0	0	0						



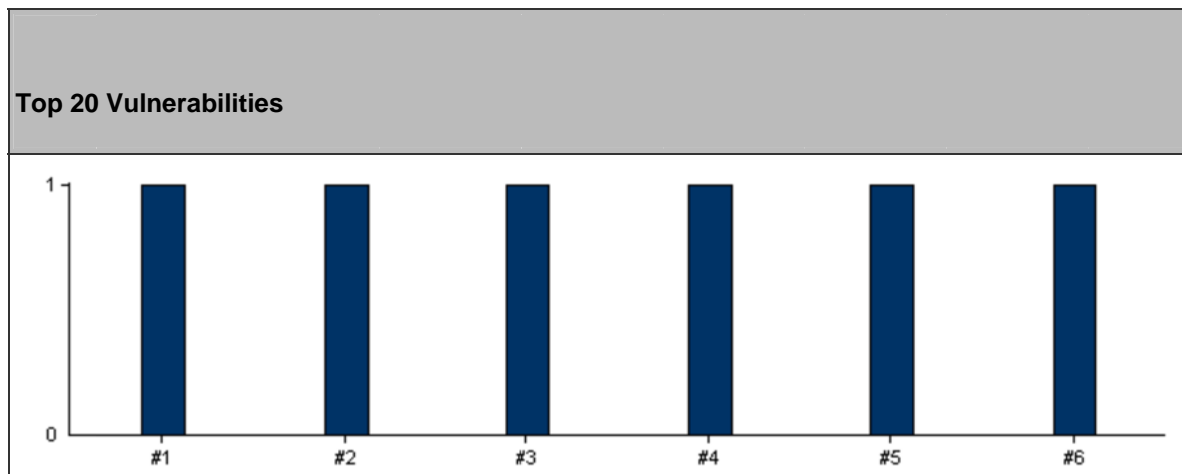






## TOP 20 VULNERABILITIES

The following is an overview of the top 20 vulnerabilities on your network.

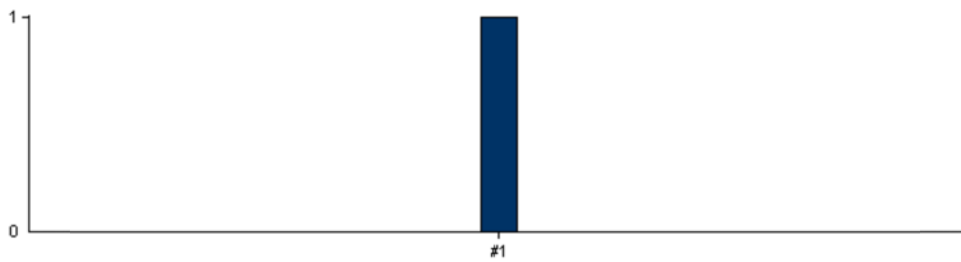


Rank	Vulnerability Name	Count
1.	ICMP Timestamp Request	1
2.	SSH Local Access Not Available	1
3.	OpenSSH X11 Port Forwarding Session Hijack Vulnerability	1

## TOP 20 OPEN PORTS

The following is an overview of the top 20 open ports on your network.

Rank	Port Number	Description	Count
1.	TCP:22	SSH - SSH (Secure Shell) Remote Login Protocol	1



### TOP 20 RUNNING SERVICES

The following is an overview of the top 20 running services on your network.

Rank	Name	Description	Count
No Services Discovered			

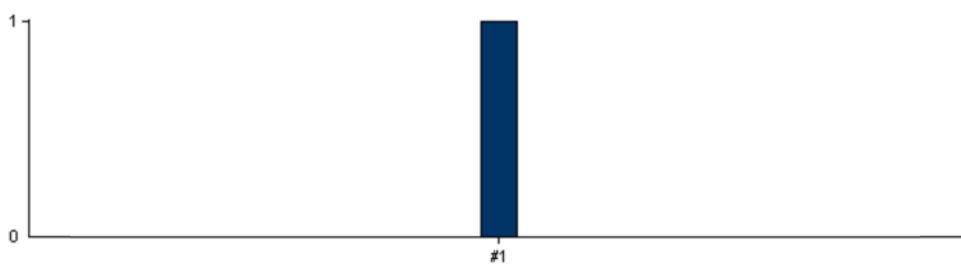
Top 20 Running Services			
No Services Discovered			

### TOP 20 OPERATING SYSTEMS

The following is an overview of the top 20 operating systems on your network.

Rank	Operating System Name	Count
1.	Linux 2.4.6 - 2.4.21	1

Top 20 Operating Systems			
--------------------------	--	--	--



### TOP 20 USER ACCOUNTS

The following is an overview of the top 20 user accounts on your network.

Rank	Account Name	Count
No Users Discovered		
<b>Top 20 User Accounts</b>		
No Users Discovered		

### TOP 20 NETWORK SHARES

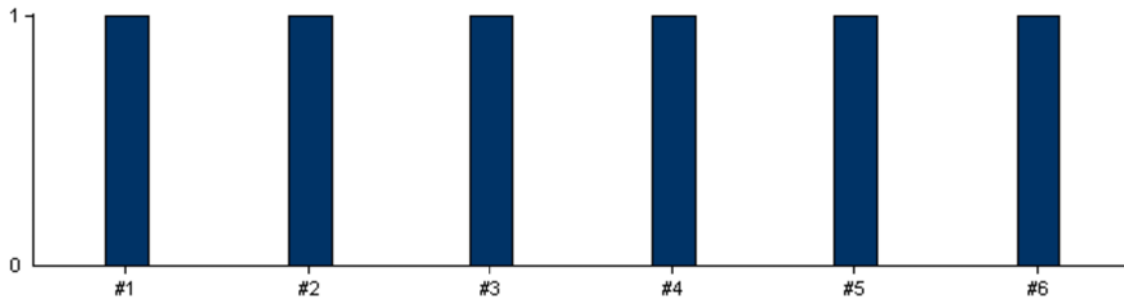
The following is an overview of the top 20 network shares on your network.

Rank	Share Name	Count
No Shares Discovered		
<b>Top 20 Network Shares</b>		
No Shares Discovered		

### BOTTOM 20 VULNERABILITIES

The following is an overview of the bottom 20 vulnerabilities on your network.

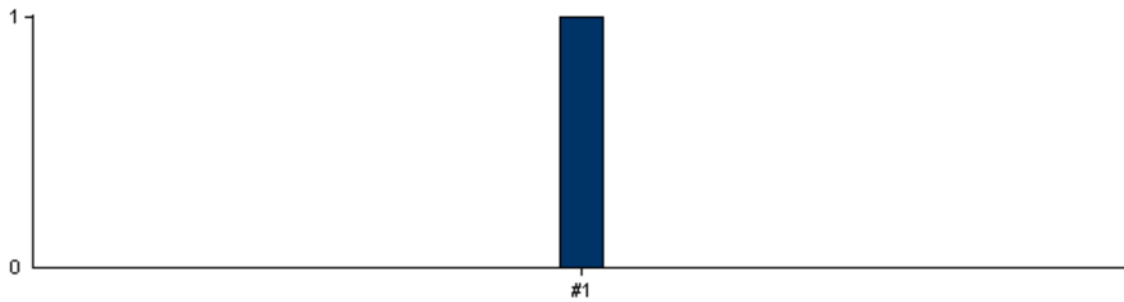
Rank	Vulnerability Name	Count
1.	ICMP Timestamp Request	1
2.	SSH Local Access Not Available	1
3.	OpenSSH X11 Port Forwarding Session Hijack Vulnerability	1
<b>Bottom 20 Vulnerabilities</b>		



### BOTTOM 20 OPEN PORTS

The following is an overview of the bottom 20 open ports on your network.

Rank	Port Number	Description	Count
1.	TCP:22	SSH - SSH (Secure Shell) Remote Login Protocol	1



### BOTTOM 20 RUNNING SERVICES

The following is an overview of the bottom 20 running services on your network.

Rank	Name	Description	Count
No Services Discovered			

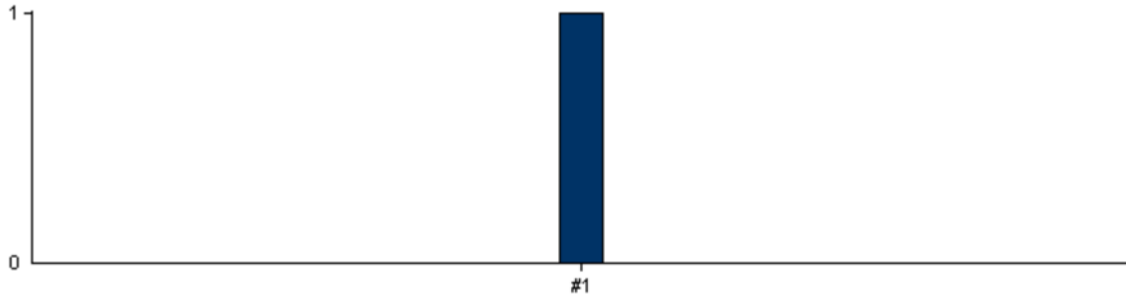
Bottom 20 Running Services
No Services Discovered

### BOTTOM 20 OPERATING SYSTEMS

The following is an overview of the bottom 20 operating systems on your network.

Rank	Operating System Name	Count
1.	Linux 2.4.6 - 2.4.21	1

#### Bottom 20 Operating Systems



### BOTTOM 20 USER ACCOUNTS

The following is an overview of the bottom 20 user accounts on your network.

Rank	Account Name	Count
No Users Discovered		

Bottom 20 User Accounts
No Users Discovered

## WINDOWS XP SCAN REPORT



eEye Digital Security™ **Retina - Network Security Scanner**  
*Network Vulnerability Assessment & Remediation Management*

07/03/2010 - Report created by Retina version 5.11.3.2195

### Metrics for 'FULL SCAN'

File name:	D:\Program Files\eEye Digital Security\Retina 5\Scans\XPWINDOWS.rtd
Audits revision:	2195
Scanner version:	5.11.3
Start time:	07/03/2010 01:10:54
Duration:	0d 0h 5m 14s
Credentials:	CCFBD62133E24AFF98183FA10417F764

<b>Audit groups:</b>	All Audits
<b>Address groups:</b>	N/A
<b>IP ranges:</b>	192.168.1.3
<b>Total hosts attempted:</b>	1
<b>Total hosts scanned:</b>	1
<b>No access:</b>	0

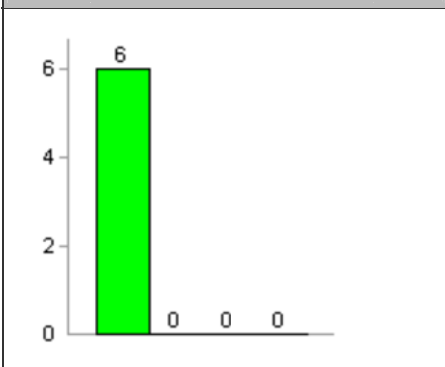
<b>NETWORK ANALYSIS RESULTS</b>			
<b>Report Summary</b>			
Scanner	Retina	Machines	1
Name		Scanned	
Scanner	5.11.3.2195	Vulnerabilities	0
Version		Total	
Scan Start	07/03/2010	High Risk	0
Date		Vulnerabilities	
Scan Start	01:10:54	Medium Risk	0
Time		Vulnerabilities	
Scan	0h 5m 14s	Low Risk	0
Duration		Vulnerabilities	
Scan	FULL	Information	6
Name	SCAN	Only Audits	
Scan	Completed	Credential	CCFBD62133E24AFF98183FA10417F764

Status	Used
Vulnerable Machines	1

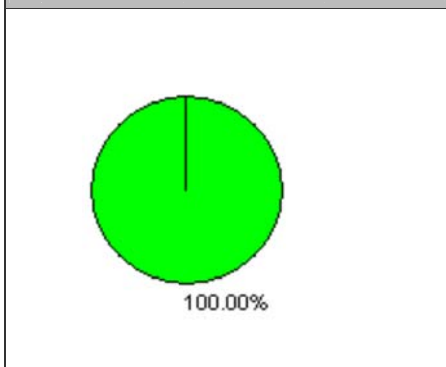
### Top 5 Most Vulnerable Hosts



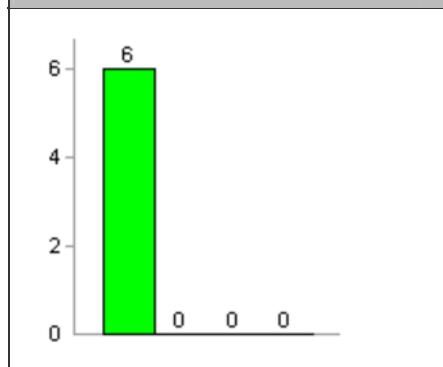
### Num. of Vulnerabilities By Risk



### % of Vulnerabilities By Risk



### Avg. of Vulnerabilities By Risk

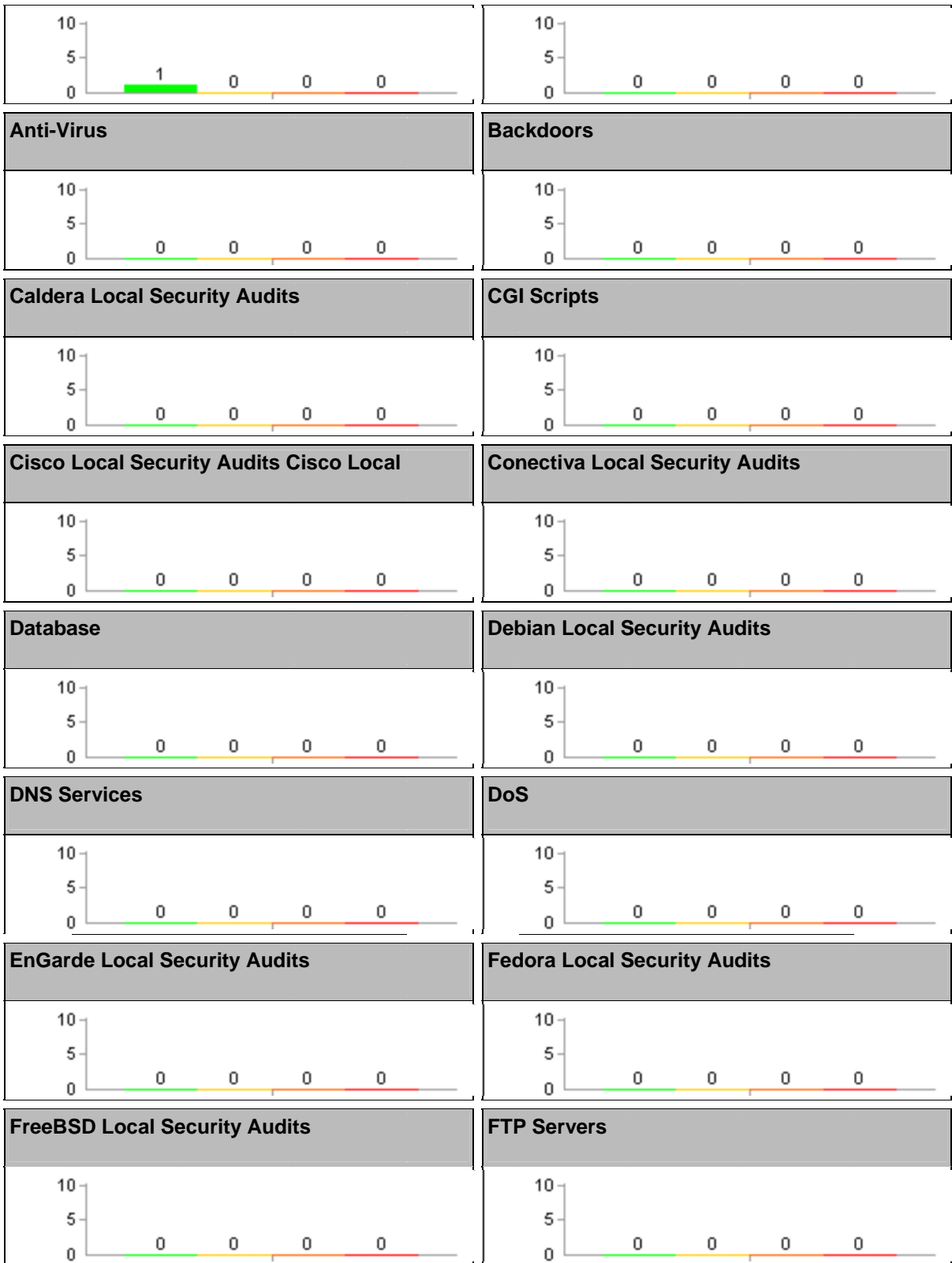


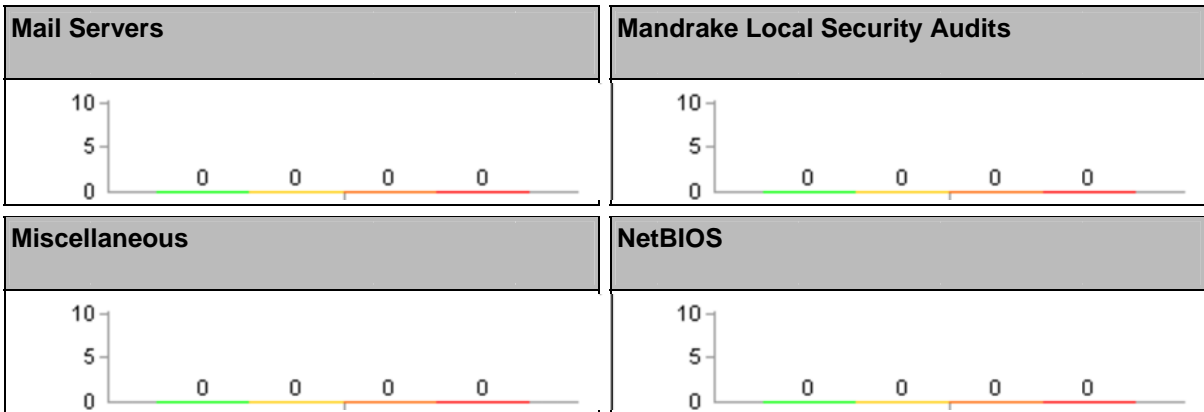
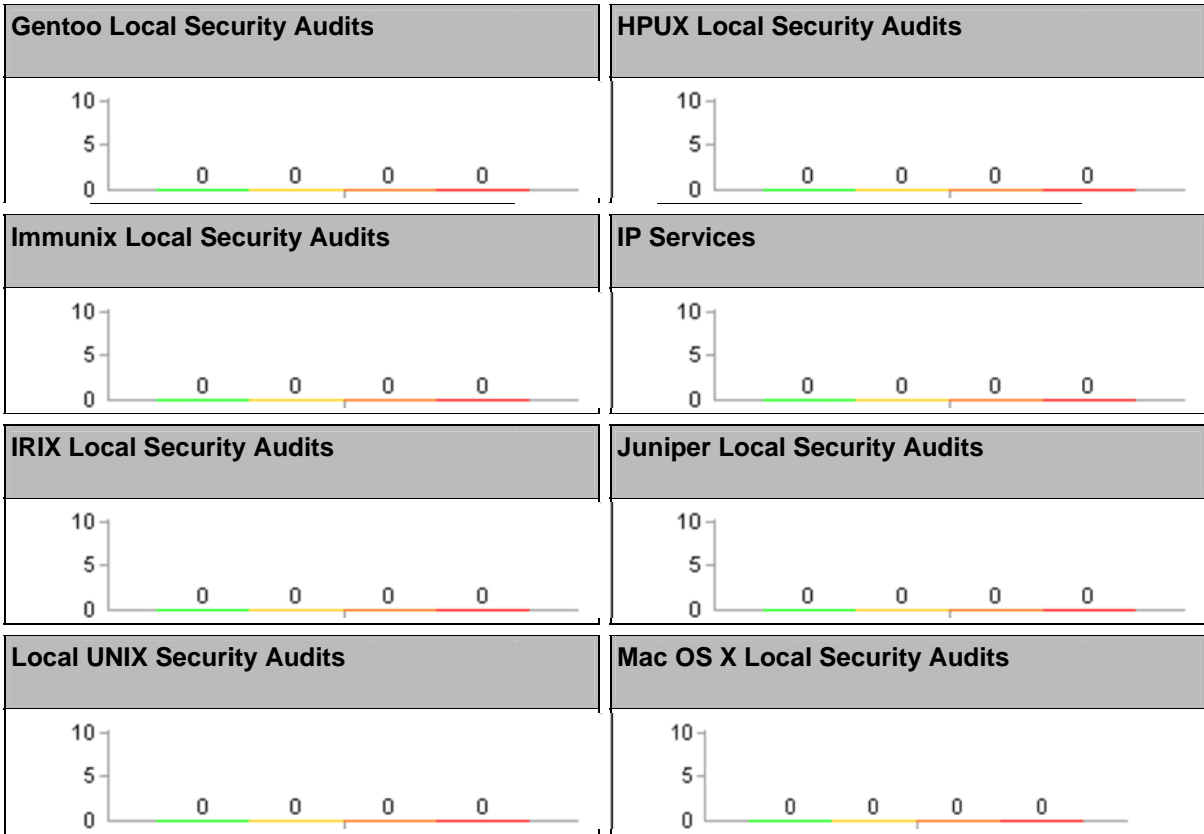
## TOTAL VULNERABILITIES BY CATEGORY

The following is an overview of the total vulnerabilities by audit category.

Accounts	AIX Local Security Audits
----------	---------------------------









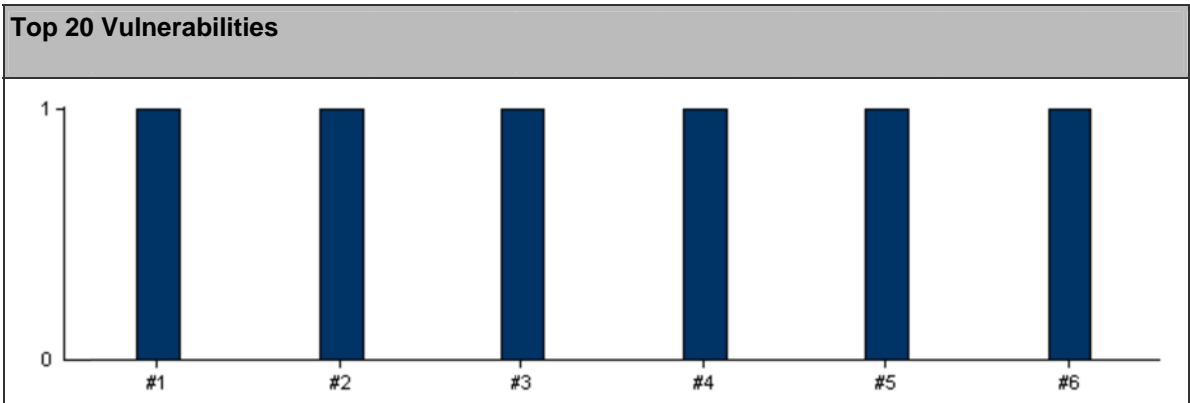


### TOP 20 VULNERABILITIES

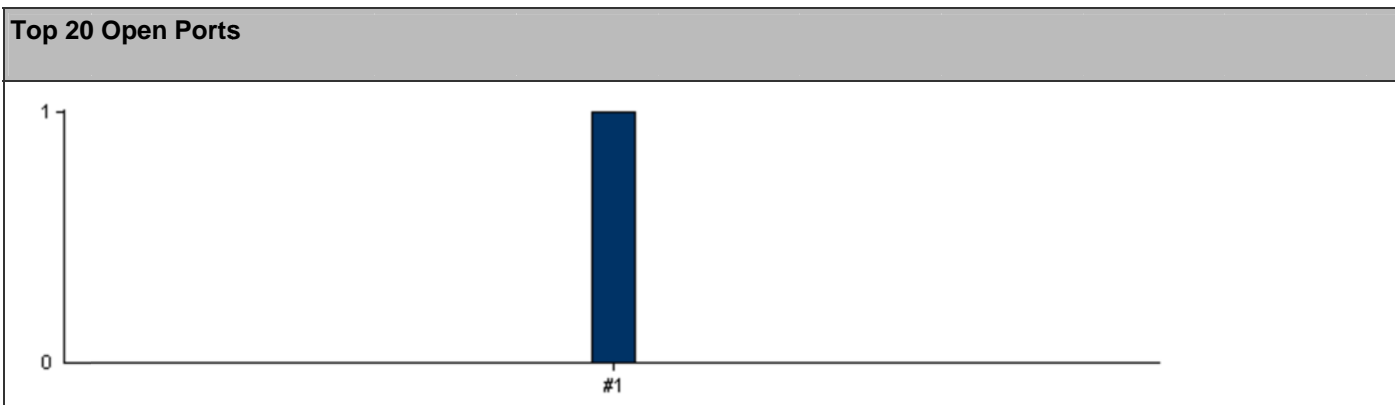
The following is an overview of the top 20 vulnerabilities on your network.

Rank	Vulnerability Name	Count
1.	Verify Microsoft Windows Non-Default User Services	1
2.	Verify Microsoft Windows Anonymous SID/Name Translation	1
3.	Verify Microsoft Windows Password Complexity	1

4.	Verify Microsoft Windows Users with Administrative Privileges	1
5.	Verify Microsoft Windows Users with Backup Operator Privileges	1
6.	Verify Software Certificate Installation Files	1



TOP 20 OPEN PORTS		Port Number	Description	Count
The following is an overview of the top 20 open ports on your network. Rank				
1.		TCP:9424		1



**TOP 20 RUNNING SERVICES**

The following is an overview of the top 20 running services on your network.

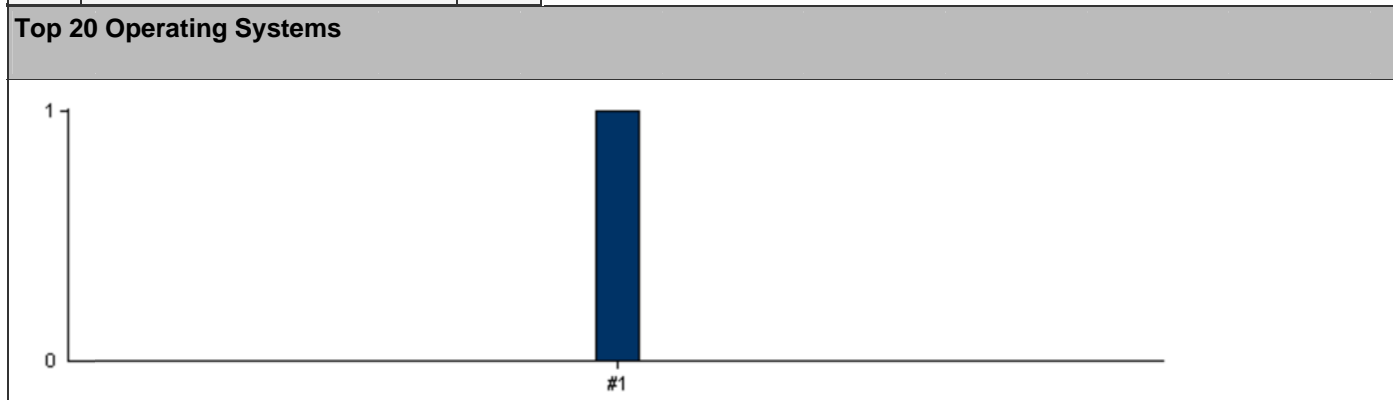
Rank	Name	Description	Count
No Services Discovered			

Top 20 Running Services
No Services Discovered

### TOP 20 OPERATING SYSTEMS

The following is an overview of the top 20 operating systems on your network.

Rank	Operating System Name	Count
1.	Microsoft Windows Server 2003	1



### TOP 20 USER ACCOUNTS

The following is an overview of the top 20 user accounts on your network.

Rank	Account Name	Count
No Users Discovered		

Top 20 User Accounts
No Users Discovered

#### TOP 20 NETWORK SHARES

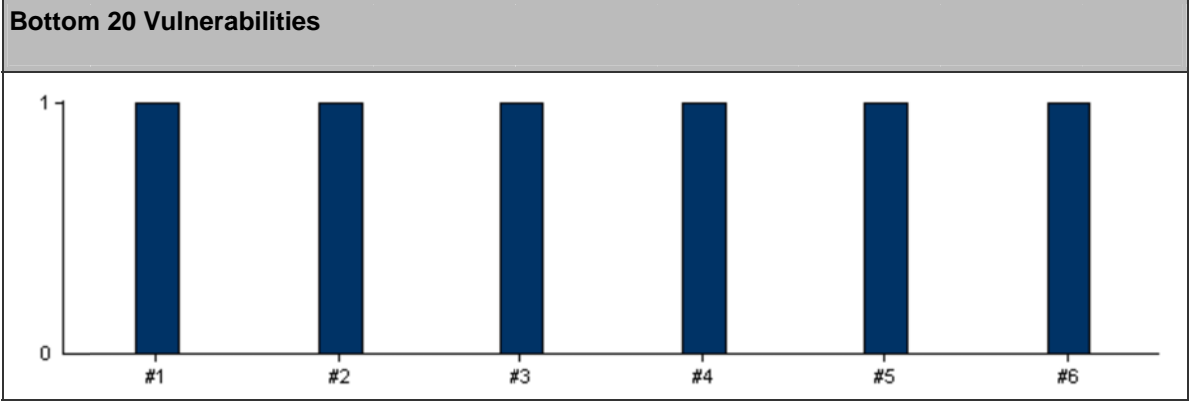
The following is an overview of the top 20 network shares on your network.

Rank	Share Name	Count
No Shares Discovered		
Top 20 Network Shares		
No Shares Discovered		

#### BOTTOM 20 VULNERABILITIES

The following is an overview of the bottom 20 vulnerabilities on your network.

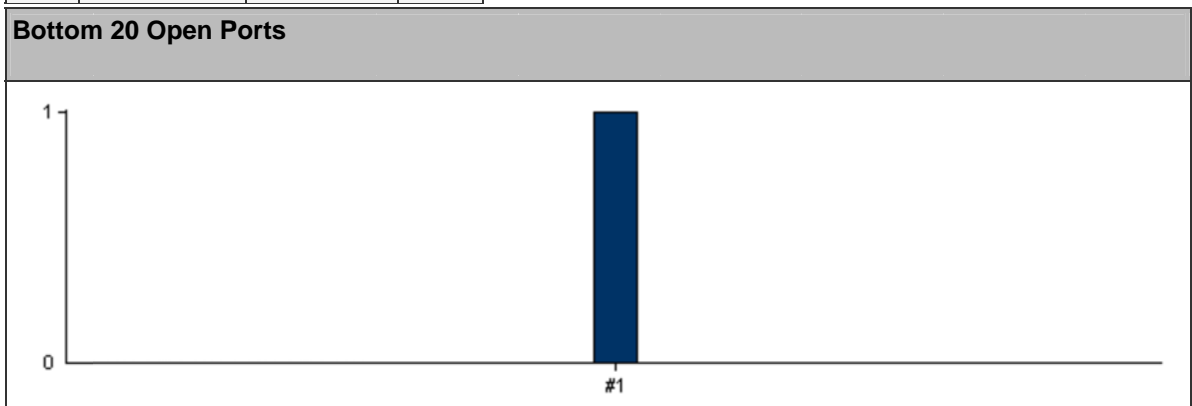
Rank	Vulnerability Name	Count
1.	Verify Microsoft Windows Non-Default User Services	1
2.	Verify Microsoft Windows Anonymous SID/Name Translation	1
3.	Verify Microsoft Windows Password Complexity	1
4.	Verify Microsoft Windows Users with Administrative Privileges	1
5.	Verify Microsoft Windows Users with Backup Operator Privileges	1
6.	Verify Software Certificate Installation Files	1



### BOTTOM 20 OPEN PORTS

The following is an overview of the bottom 20 open ports on your network.

Rank	Port Number	Description	Count
1.	TCP:9424		1



### BOTTOM 20 RUNNING SERVICES

The following is an overview of the bottom 20 running services on your network.

Rank	Name	Description	Count
No Services Discovered			

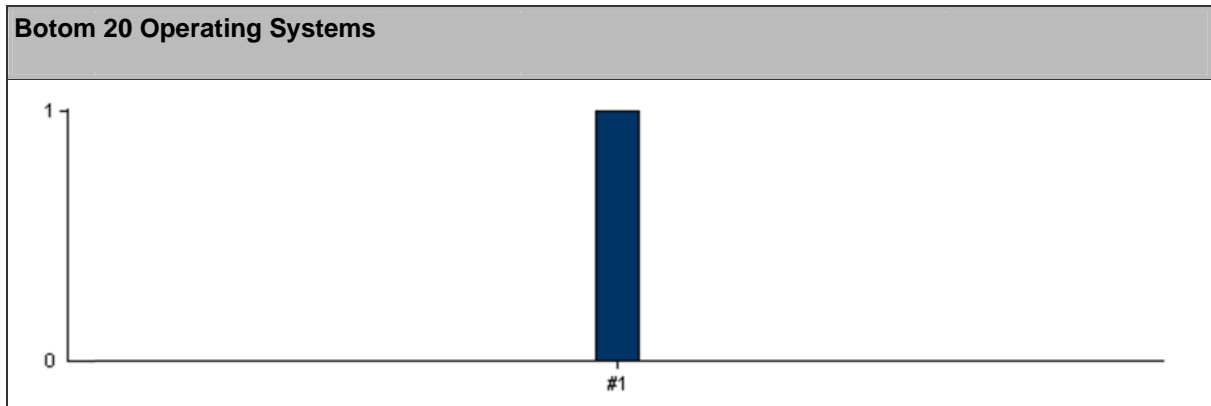


Bottom 20 Running Services
No Services Discovered

**BOTTOM 20 OPERATING SYSTEMS**

The following is an overview of the bottom 20 operating systems on your network.

Rank	Operating System Name	Count
1.	Microsoft Windows Server 2003	1



**BOTTOM 20 USER ACCOUNTS**

The following is an overview of the bottom 20 user accounts on your network.

Rank	Account Name	Count
No Users Discovered		

Bottom 20 User Accounts
No Users Discovered

## BOTTOM 20 NETWORK SHARES

The following is an overview of the bottom 20 network shares on your network.

Rank
Share Name
Count
No Shares Discovered

## GLOSSARY

The following is glossary of common terms used throughout this report.

- **DoS Attack:** A Denial of Service (DoS) attack is a remote attack against a servers TCP/IP stack or services. DoS attacks can saturate a servers bandwidth, saturate all available connections for a particular service, or even crash a server.
- **Exploit:** A script or program that takes advantage of vulnerabilities in services or programs to allow an attacker to gain unauthorized or elevated system access.
- **Host:** A node on a network. Usually refers to a computer or device on a network which both initiates and accepts network connections.
- **IP Address:** The 32-bit address defined by the Internet Protocol in STD 5, RFC 791. It is usually represented in dotted decimal notation. Any device connected to the Internet that used TCP/IP is assigned an IP Address. An IP Address can be likened to a home address in that no two are

alike.

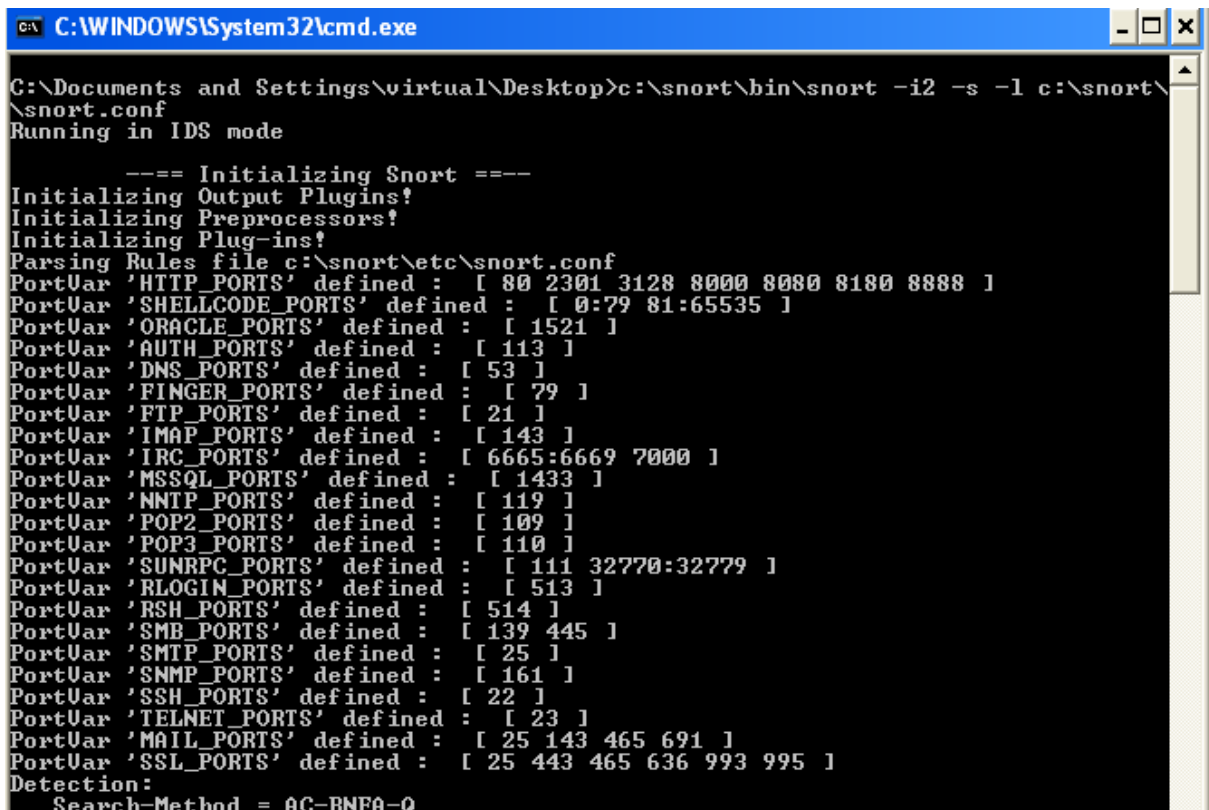
- **Netbios:** Network Basic Input Output System. The standard interface to networks on IBM PC and compatible networks.
- **Ping:** A program used to test reachability of destination nodes by sending them an ICMP echo request and waiting for a reply.
- **Port:** A port in the network sense is the pathway that a computer uses to transmit and receive data. As an example, Web Servers typically listen for requests on port 80.
- **Registry:** The internal system configuration that a user can customize to alter his computing environment on the Microsoft Windows Platform. The registry is organized in a hierarchical structure of subtrees and their respective keys, subkeys, and values that apply to those keys and subkeys
- **Risk Level - Info:** Retina may provide additional information about a host that does not necessarily represent a security threat, but may be useful to the administrator in order to better assess the security of the host, or the network at large. These alerts are displayed with the list of discovered vulnerabilities, and are indicated by a green 'i' icon.
- **Risk Level - Low:** A low-risk vulnerability is typically one that only presents a threat in specific and unlikely circumstances. Such a vulnerability may provide an attacker with information that could be combined with other, higher-risk vulnerabilities, in order to compromise the host or its users.
- **Risk Level - Medium:** Medium-risk vulnerabilities are serious security threats that would allow a trusted but non-privileged user to assume complete control of a host, or would permit an untrusted user to disrupt service or gain access to sensitive information.

- **Risk Level - High:** A vulnerability is designated as high-risk if it would allow a user who has not been given any amount of trust on a susceptible host to take control of it. Other vulnerabilities that severely impact the overall safety and usability of the network may also be designated as high-risk.
- **Service:** A service is a program running on a remote machine that in one way or another provides a service to users. For example, when you visit a website the remote server displays a web page via its web server service.
- **Share:** A folder, set of files, or even a hard drive partition set up on a machine to allow access to other users. Shares are frequently set up with incorrect file permissions which could allow an attacker to gain access to this data.
- **Sniffer:** frequently attackers will place a sniffer program on a compromised machine. The sole purpose of a sniffer is to collect data being transmitted on the network in clear-text including usernames and passwords.
- **Subnet:** A portion of a network, which may be a physically independent network segment, which shares a network address with other portions of the network and is distinguished by a subnet number.
- **Vulnerability:** A weakness or a flaw in a program or service that can allow an attacker to gain unauthorized or elevated system access.

## Appendix D

### SNORT STAT AND LOGS

In Chapter 4 reference was made to examples of a SNORT run. SnortAlog is a Perl script that reads SNORT alert logs, firewall logs and generates a report in PDF, html or text format. It summarises alerts on three features: alert type, source and destination address. Through this analysis we can see the distribution of attack methods, types of alerts and percentage of attacks against each host. The example source is from the experimental setup.



```
C:\WINDOWS\System32\cmd.exe
C:\Documents and Settings\virtual\Desktop>c:\snort\bin\snort -i2 -s -l c:\snort\snort.conf
Running in IDS mode

    ---= Initializing Snort =---
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file c:\snort\etc\snort.conf
PortUar 'HTTP_PORTS' defined : [ 80 2301 3128 8000 8080 8180 8888 ]
PortUar 'SHELLCODE_PORTS' defined : [ 0:79 81:65535 ]
PortUar 'ORACLE_PORTS' defined : [ 1521 ]
PortUar 'AUTH_PORTS' defined : [ 113 ]
PortUar 'DNS_PORTS' defined : [ 53 ]
PortUar 'FINGER_PORTS' defined : [ 79 ]
PortUar 'FTP_PORTS' defined : [ 21 ]
PortUar 'IMAP_PORTS' defined : [ 143 ]
PortUar 'IRC_PORTS' defined : [ 6665:6669 7000 ]
PortUar 'MSSQL_PORTS' defined : [ 1433 ]
PortUar 'NNTP_PORTS' defined : [ 119 ]
PortUar 'POP2_PORTS' defined : [ 109 ]
PortUar 'POP3_PORTS' defined : [ 110 ]
PortUar 'SUNRPC_PORTS' defined : [ 111 32770:32779 ]
PortUar 'RLOGIN_PORTS' defined : [ 513 ]
PortUar 'RSH_PORTS' defined : [ 514 ]
PortUar 'SMB_PORTS' defined : [ 139 445 ]
PortUar 'SMTP_PORTS' defined : [ 25 ]
PortUar 'SNMP_PORTS' defined : [ 161 ]
PortUar 'SSH_PORTS' defined : [ 22 ]
PortUar 'TELNET_PORTS' defined : [ 23 ]
PortUar 'MAIL_PORTS' defined : [ 25 143 465 691 ]
PortUar 'SSL_PORTS' defined : [ 25 443 465 636 993 995 ]
Detection:
Search-Method = AC-BNFA-Q
```

```

[ Port and Service Based Pattern Matching Memory ]
+---[AC-BNFA Search Info Summary]-----+
| Instances      : 961
| Patterns       : 261835
| Pattern Chars  : 2586220
| Num States     : 1567535
| Num Match States : 233170
| Memory         : 37.97Mbytes
|   Patterns     : 7.46M
|   Match Lists  : 11.60M
|   Transitions  : 18.69M
+-----+

      ---= Initialization Complete =---

o'~)~
''''~
-*> Snort! <*-
Version 2.8.3.1-ODBC-MySQL-FlexRESP-WIN32 GRE <Build 17>
By Martin Roesch & The Snort Team: http://www.snort.org/team.html
<C> Copyright 1998-2008 Sourcefire Inc., et al.
Using PCRE version: 7.4 2007-09-21

Rules Engine: SF_SNORT_DETECTION_ENGINE Version 1.9 <Build 15>
Preprocessor Object: SF_SSH Version 1.1 <Build 1>
Preprocessor Object: SF_SMTP Version 1.1 <Build 7>
Preprocessor Object: SF_FTPTELNET Version 1.1 <Build 10>
Preprocessor Object: SF_DNS Version 1.1 <Build 2>
Preprocessor Object: SF_DCERPC Version 1.1 <Build 4>

```

Please see next page

File Edit View Manage Help

Display 00 (Default)

!	Date	Time	Priority	Hostname	Message
	03-07-2010	00:59:26	Auth.Alert	127.0.0.1	snort: [1:480:6] ICMP PING speedera [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.3
	03-07-2010	00:59:26	Auth.Alert	127.0.0.1	snort: [1:384:5] ICMP PING [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.3
	03-07-2010	00:59:20	Auth.Alert	127.0.0.1	snort: [1:480:6] ICMP PING speedera [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.3
	03-07-2010	00:59:20	Auth.Alert	127.0.0.1	snort: [1:384:5] ICMP PING [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.3
	03-07-2010	00:55:36	Auth.Alert	127.0.0.1	snort: [1:402:8] ICMP Destination Unreachable Port Unreachable [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.1
	03-07-2010	00:53:52	Auth.Alert	127.0.0.1	snort: [1:486:5] ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.2
	03-07-2010	00:53:52	Auth.Alert	127.0.0.1	snort: [1:5897:3] SPYWA [Classification: Misc activity] [Priority: 3]: {UDP} 192.168.1.2:3069 -> 192.168.1.7:407
	03-07-2010	00:53:49	Auth.Alert	127.0.0.1	snort: [1:486:5] ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.2
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:3626:4] ICMP PA [Classification: Misc activity] [Priority: 2]: {ICMP} 192.168.1.2 -> 192.168.1.7
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:396:7] ICMP Des [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.7
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:402:8] ICMP Des [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.7
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:404:7] ICMP Des [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.7
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:451:5] ICMP Tim [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.2
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:451:5] ICMP Tim [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.2
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:453:5] ICMP Tim [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.7
	03-07-2010	00:53:45	Auth.Alert	127.0.0.1	snort: [1:453:5] ICMP Tim [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.2 -> 192.168.1.7
	03-07-2010	00:53:43	Auth.Alert	127.0.0.1	snort: [1:486:5] ICMP Des [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.2
	03-07-2010	00:53:39	Auth.Alert	127.0.0.1	snort: [1:486:5] ICMP Des [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.2
	03-07-2010	00:53:39	Auth.Alert	127.0.0.1	snort: [1:2049:6] SQL pin [Classification: Misc activity] [Priority: 3]: {TCP} 192.168.1.2:3065 -> 192.168.1.7:1434
	03-07-2010	00:53:34	Auth.Alert	127.0.0.1	snort: [1:486:5] ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.2
	03-07-2010	00:53:34	Auth.Alert	127.0.0.1	snort: [1:12198:4] SNMP MS Windows getbulk request [Classification: Attempted Administrator Privilege Gain] [Priority: 1]: {UDP} 192.168.1.2:3064 -> 192.168.1.7:161
	03-07-2010	00:53:33	Auth.Alert	127.0.0.1	snort: [1:486:5] ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.2
	03-07-2010	00:53:33	Auth.Alert	127.0.0.1	snort: [1:486:5] ICMP Destination Unreachable Communication with Destination Host is Administratively Prohibited [Classification: Misc activity] [Priority: 3]: {ICMP} 192.168.1.7 -> 192.168.1.2
	03-07-2010	00:53:27	Auth.Alert	127.0.0.1	snort: [1:1413:13] SNMP private access udp [Classification: Attempted Information Leak] [Priority: 2]: {UDP} 192.168.1.2:3063 -> 192.168.1.7:161

Syslog Statistics

History (1hr) History (24hr) Severity Top 20 Hosts Counters

Help (F1) Refresh (F5) Close

## Appendix E

This appendix gives the code for designing the system on level 3, 2, and 1 respectively as referred to in Chapter 5 of this thesis. The working model is designed using Matlab, where there are 18 membership rules in terms of the likelihood of fuzzy unions and intersections considered for this particular case. The AF results are analysed for each logical level in the tree. Mamdani's Method was adopted because of ease of use and implementation. For a given value of each node the AF activity graph varies between unlikely and likely for the input range of 0 and 100. The code is shown below.

### Codes for designing the system. LEVEL 3

```
[System]
Name='Thesisexperiment2010final'
Type='mamdani'
Version=2.0
NumInputs=4
NumOutputs=1
NumRules=6
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='NODE:D-r-pr'
Range=[0 100]
NumMFs=3
MF1='LESS':'gausmf',[15 0]
MF2='AVERAGE':'gausmf',[15 50]
MF3='HIGH':'gausmf',[15 100]

[Input2]
Name='NODE:D-r-mr'
Range=[0 100]
NumMFs=3
MF1='LESS':'gausmf',[21.23 -4.441e-016]
MF2='AVERAGE':'gausmf',[21.23 50]
MF3='HIGH':'gausmf',[21.23 100]

[Input3]
```



```
Name='NODE:D-r-ur'  
Range=[0 100]  
NumMFs=3  
MF1='LESS': 'gaussmf', [21.23 -4.441e-016]  
MF2='AVERAGE': 'gaussmf', [21.23 50]  
MF3='HIGH': 'gaussmf', [21.23 100]
```

[Input4]

```
Name='NODE:D-r-kr'  
Range=[0 100]  
NumMFs=3  
MF1='LESS': 'gaussmf', [21.23 -4.441e-016]  
MF2='AVERAGE': 'gaussmf', [21.23 50]  
MF3='HIGH': 'gaussmf', [21.23 100]
```

[Output1]

```
Name='AFRESULT'  
Range=[0 100]  
NumMFs=3  
MF1='LESS': 'trimf', [0 16.67 33.33]  
MF2='AVERAGE': 'trimf', [33.33 50 66.67]  
MF3='HIGH': 'trimf', [66.67 83.33 100]
```

[Rules]

```
1 0 0 0, 1 (1) : 2  
2 0 0 0, 2 (1) : 1  
3 0 0 0, 3 (1) : 2  
1 0 0 0, 1 (1) : 2  
2 0 0 0, 1 (1) : 2  
3 0 0 0, 1 (1) : 2
```

## Codes for designing the system. LEVEL 2

[System]

```
Name='Thesisexperiment2010finalNode2'  
Type='mamdani'  
Version=2.0  
NumInputs=6  
NumOutputs=1  
NumRules=2  
AndMethod='min'  
OrMethod='max'  
ImpMethod='min'  
AggMethod='max'  
DefuzzMethod='centroid'
```

[Input1]

```
Name='NODE:D-s'  
Range=[0 100]  
NumMFs=3  
MF1='LESS': 'gaussmf', [15 0]  
MF2='AVERAGE': 'gaussmf', [15 50]  
MF3='HIGH': 'gaussmf', [15 100]
```

```
[Input2]
Name='NODE:D-e'
Range=[0 100]
NumMFs=3
MF1='LESS': 'gausmf', [21.23 -4.441e-016]
MF2='AVERAGE': 'gausmf', [21.23 50]
MF3='HIGH': 'gausmf', [21.23 100]
```

```
[Input3]
Name='NODE:D-r'
Range=[0 100]
NumMFs=3
MF1='LESS': 'gausmf', [21.23 -4.441e-016]
MF2='AVERAGE': 'gausmf', [21.23 50]
MF3='HIGH': 'gausmf', [21.23 100]
```

```
[Input4]
Name='NODE:W-d'
Range=[0 100]
NumMFs=3
MF1='LESS': 'gausmf', [21.23 -4.441e-016]
MF2='AVERAGE': 'gausmf', [21.23 50]
MF3='HIGH': 'gausmf', [21.23 100]
```

```
[Input5]
Name='NODE:W-c'
Range=[0 100]
NumMFs=3
MF1='LESS': 'gausmf', [21.23 0]
MF2='AVERAGE': 'gausmf', [21.23 50]
MF3='HIGH': 'gausmf', [21.23 100]
```

```
[Input6]
Name='NODE:W-p'
Range=[0 100]
NumMFs=3
MF1='LESS': 'gausmf', [21.23 0]
MF2='AVERAGE': 'gausmf', [21.23 50]
MF3='HIGH': 'gausmf', [21.23 100]
```

```
[Output1]
Name='AFRESULT'
Range=[0 100]
NumMFs=3
MF1='LESS': 'trimf', [0 16.67 33.33]
MF2='AVERAGE': 'trimf', [33.33 50 66.67]
MF3='HIGH': 'trimf', [66.67 83.33 100]
```

```
[Rules]
1 1 1 1 0 0, 1 (1) : 2
1 1 1 1 0 0, 1 (1) : 1
```

## Codes for designing the system. LEVEL 1

```
[System]
Name='Thesisexperiment2010node1'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=2
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='NODE:W'
Range=[0 100]
NumMFs=3
MF1='LESS':'gausmf',[15 0]
MF2='AVERAGE':'gausmf',[15 50]
MF3='HIGH':'gausmf',[15 100]

[Input2]
Name='NODE:D'
Range=[0 100]
NumMFs=3
MF1='LESS':'gausmf',[15 0]
MF2='AVERAGE':'gausmf',[15 0]
MF3='HIGH':'gausmf',[15 0]

[Output1]
Name='AFRESULT'
Range=[0 100]
NumMFs=3
MF1='LESS':'trimf',[0 16.67 33.33]
MF2='AVERAGE':'trimf',[33.33 50 66.67]
MF3='HIGH':'trimf',[66.67 83.33 100]

[Rules]
1 1, 1 (1) : 2
1 1, 1 (1) : 1
```